Programming Support Library (PSL)
Maintenance Manual - Volume II

IBM

(9) Final rept.

50272-101

REPORT DOCUMENTATION PAGE	1. REPORT NO. (19) (18) DOD/DF-81/016e	2.	3. Recipient's Accession No. AD-A107 253
4. Title and Subtitle (6) PROGRAMMING SUPPORT LIBRARY (PSL). Maintenance Manual, Volume II.			5. Report Date (11) May 1978
7. Author(s)			8. Performing Organization Rept. No. (12) 262
9. Performing Organization Name and Address Federal Systems Division International Business Machines Corporation Gaithersburg, Maryland			10. Project/Task/Work Unit No. 11. Contract(C) or Grant(G) No. (15) F30602-77-C-0249 (C) (G)
12. Sponsoring Organization Name and Address Rome Air Development Center RADC/COEE Griffiss Air Force Base, NY 13441			13. Type of Report & Period Covered 14.

15. Supplementary Notes
(21) For magnetic tape, see AD-A107 248.
See also Volume 1, AD-A107 252.

The source agency has restricted sales of this item to Federal, state and local governments.

16. Abstract (Limit: 200 words)

The Programming Support Library (PSL) is a software system which provides the tools to organize, implement, and control computer program development. This involves the support of the actual programming process and also the support of the management process. The PSL is designed to support Top Down Design and Structured Programming (TDDSP).

This Maintenance Manual provides detailed information on the PSL program operations, data formats and special procedures to assist programmer personnel in the maintenance of PSL system programs.

17. Document Analysis a. Descriptors

Software Configuration Control
Structured Programming Support Tool
Software Development Support

b. Identifiers/Open-Ended Terms

c. COSATI Field/Group

174950

260

18. Availability Statement:	19. Security Class (This Report) UNCLASSIFIED	21. No. of Pages
	20. Security Class (This Page) UNCLASSIFIED	22. Price

(See ANSI-Z39.18) See Instructions on Reverse OPTIONAL FORM 272 (4-77)

2.2.1.3 PSL Access Routines

The modules in this subdivision of the PSL system are called to access the various data storage elements of the PSL system. The three categories of input/output (I/O) accessing are depicted in Figure 2-01 as Basic I/O, Unit I/O and Index I/O.

2.2.1.3.1 Basic I/O

Basic I/O is performed to initialize, assign and release PSL section file space and to assign, read, write and release random block storage within a section file. The program modules which perform these operations are described below.

Continuation of Volume I

2.2.1.3.1.1 ITSF - Initialize a Standard File

The ITSF module is called to initialize the random block storage that constitutes a standard section file used in PSL operations.

a. Program Operations

HIPO diagram 1.3.1.1 depicts the program operations of the ITSF modules. The first step calculates the number of PSL blocks that can be assigned within the space allocated by FMS to the section file. Rather than call the ASFL module, the ITSF module directly calls the ALFL module to allocate the section file then opens it for output. The first block (referred to as Block #0) in the section file is initialized as a PSL Control Block. The second block (referred to as Block #1) is then initialized as a PSL Index Block. If the requirement for a file directory is indicated by the FILE-CONTAINS-DIRECTORY-SW value, then the third block (referred to as Block #2) is initialized as a PSL Index Block for use as a file directory. The PSL Control Block is initialized with block number pointers to the index block (Block #1) and the directory block (Block #2) when required, else the directory block pointer will have a zero value. The PSL Control Block is also initialized with the library section name, the section password (if any) and the section options. Most importantly, the control block is initialized with a one dimensional string of free-block-switch-bit values in which a one-bit indicates that the block number for a correspondingly indexed (i.e., numbered) bit position is free for block assignment in PSL operations and a zero-bit indicates that the block is not free for assignment; otherwise stated: it is busy. If the last block number that can be assigned within the section is numerically greater than the number of free-block-switch bit positions available in the first control block (Block #0), continuation-control blocks are required to provide as many free-block-switch bit positions as needed to indicate the free block status of the last block that may be assigned. It should be noted at this point that no free-block-switch corresponds to Block #0; that is, the first free-block-switch corresponds to Block #1.

Initialization of the first control block is performed to turn on the free block switches for all assignable block positions and then to turn off the free block switches for that number of blocks which are determined to be used; that is, the continuation-control blocks (when needed), the file directory block (when used) and unit index block (which is always used). The initialization of the index

and data blocks is performed to set the data contents to zeroes (or low values). Initialization of the continuation control blocks is performed to turn on the free block switches up to the last-block bit position. Unassignable block-number bit positions in the final control block remain set to zero indicating their busy (i.e., unavailable) status. Finally, the section file is closed and de-allocated to conclude program operations.

b. Data File and Table Descriptions

The following data tables have special use in ITSF module program operations:

```
01  PSL-CONTROL-BLOCK.
    05  CB-BLOCK-NBR                    PIC S9(9) COMP.
    05  CB-NEXT-BLOCK-NBR               PIC S9(9) COMP.
    05  CB-MAIN-DATA.
        10  CB-SECTION-DATA             PIC X(54).
        10  FILLER                      PIC X(702).
    05  CB-FREE-BLOCK-SWS   REDEFINES CB-MAIN-DATA.
        10  CB-FREE-BLOCK-SW            OCCURS 756 TIMES
                                        PIC X.
            88 CB-NO-FREE-BLOCKS-IN-SW     VALUE ZERO.
```

The above PSL-CONTROL-BLOCK description abbreviates the complete description contained in the PSL Copy Library (CPYLIB). That is, CB-SECTION-DATA is used here to denote a fuller description in the Copy Library of the individual items of data which identify the section name, section password, etc. The important aspect of the above description is noted in the redefinition of CB-MAIN-DATA by CB-FREE-BLOCK-SWS.

In the first control block, fifty-four (54) of the redefined table positions are utilized for section data. This fact indicates the need to initialize the free-block-sw-pointer to fifty-four when processing the free block switches in the first control block as compared with an initial setting of zero when processing the switches in continuation control blocks.

```
01  ON-BIT-PATTERN-VALUES.
    05  FILLER                  PIC S9(9) COMP VALUE 32.
    05  FILLER                  PIC S9(9) COMP VALUE 48.
    05  FILLER                  PIC S9(9) COMP VALUE 56.
    05  FILLER                  PIC S9(9) COMP VALUE 60.
    05  FILLER                  PIC S9(9) COMP VALUE 62.
01  FILLER REDEFINES ON-BIT-PATTERN-VALUES.
```

```
    05  FILLER                     OCCURS 5 TIMES.
        10  FILLER                 PIC X(5).
        10  ON-BIT-PATTERN         PIC X.
```

The preceding table is defined to contain patterns of one-bits which correspond to 1, 2, 3, 4 and 5 free switch position out of the six represented in a BCD character bit configuration. For example, a decimal 32 is equivalent to an octal 40 which is equivalent to a binary 100000, and so on through decimal 62 equivalent to a binary 111110. These bit patterns are used to set the last free block switch if the last block number divided by the number of bits per character (i.e., 6) produces a non-zero remainder. The calculated, non-zero remainder is used as an index to the ON-BIT-PATTERN required to set the last free block switch.

```
01  OFF-BIT-PATTERN-VALUES.
    05                          PIC S9(9) COMP VALUE 31.
    05                          PIC S9(9) COMP VALUE 15.
    05                          PIC S9(9) COMP VALUE 7.
    05                          PIC S9(9) COMP VALUE 3.
    05                          PIC S9(9) COMP VALUE 1.
01  FILLER REDEFINES OFF-BIT-PATTERN-VALUES.
    05  FILLER                 OCCURS 5 TIMES.
        10  FILLER             PIC X(5).
        10  OFF-BIT-PATTERN PIC X.
```

In a similar way the OFF-BIT-PATTERN-VALUES table contains patterns of zero-bits in the OFF-BIT-PATTERN character that correspond to 1, 2, 3, 4 and 5 busy block switch settings. For example, decimal 31 corresponds to binary 011111 denoting that the first indicated block is assigned, and so on. The needed off bit pattern is determined from the non-zero remainder resulting from a division of the last-block-number-used by six where the quotient represents the number of full (i.e., six-bit character) switches to be turned off (i.e., made zero).

c. Branching and Error Conditions

The following branching and error conditions apply to HIPO diagram 1.3.1.1:

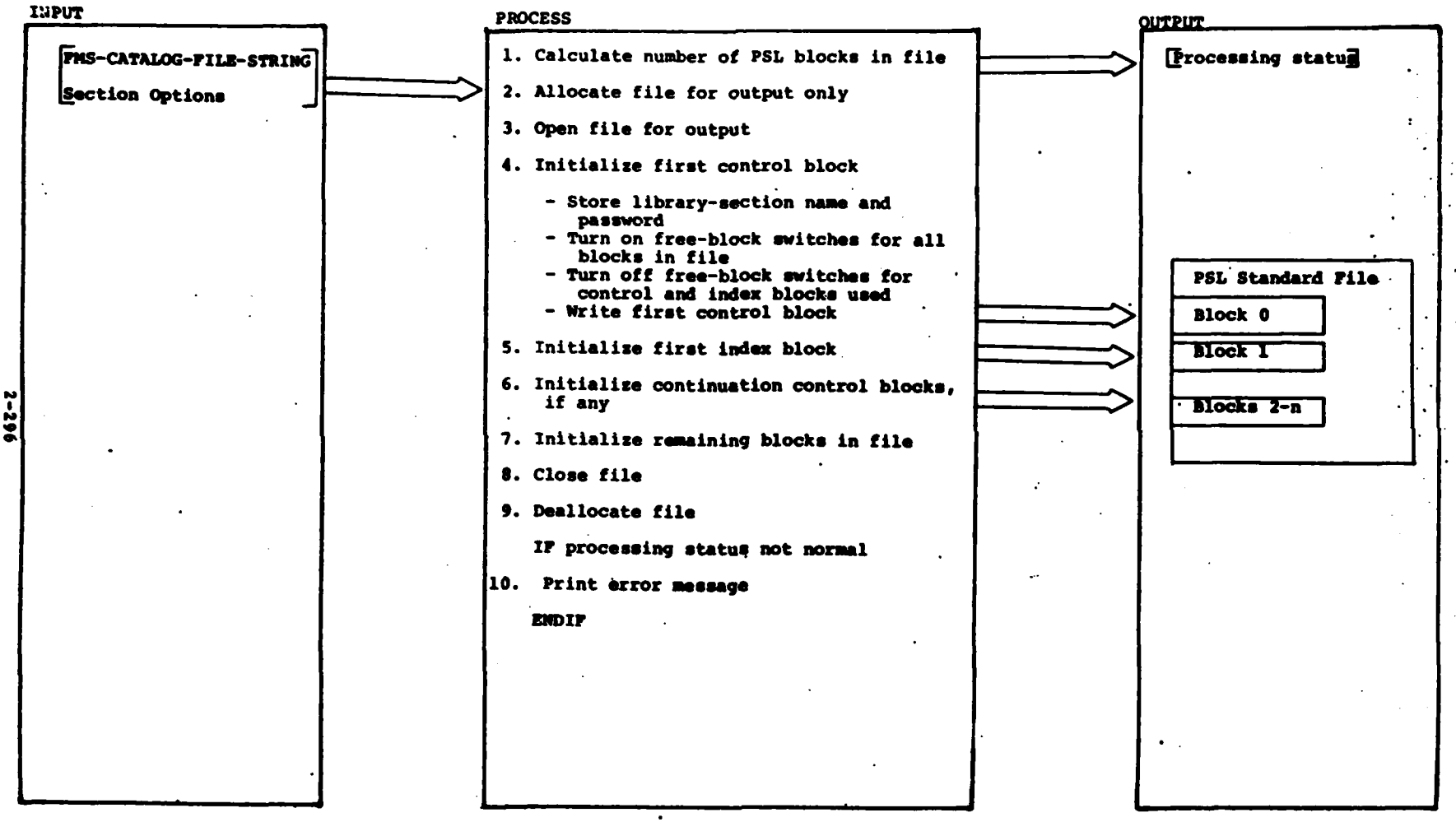
Function Reference	Condition Code	Message Category	Program Action	Note
4-7	21	PSL	Perform Process #10	1
10	23	ERR	Error exit	2

NOTES:

(1) This error is not expected to occur in normal PSL operations.

(2) This error condition is determined only when the preceding PSL error occurs.

Diagram ID: 1.3.1.1 Name: ITSF - Initialize a Standard File Description: PSL Access Routine (Basic I/O)



2.2.1.3.1.2 ACFL - Access a File

The ACFL module dynamically controls the allocation and deallocation of all files in PSL libraries. It has available four file codes for random files, two file codes for sequential input files and one file code for a sequential output file. At the request of the calling program (usually a function processor), through a call to ASFL, a PSL library file is allocated to one of these file codes of an appropriate type. The file remains allocated to the selected file code until it is released with a call to release a file (RLFL) and that file code is required for the allocation of another PSL library file. At the conclusion of all processing for a PSL run, module BCTL calls Release All Files (RLAF) to deallocate all files which have been allocated but not yet deallocated. In this way, a single PSL execution can make use of any number of PSL library files as long as concurrent use of more than four random and three sequential files is not required. Repetitive allocation and deallocation of the same file when it is used by a number of consecutive PSL functions is avoided by delaying the deallocation of file until the file code it occupies is actually required for another file. Each function processor or any other module which calls ASFL to allocate a PSL library file is expected to call RLFL to release that file when it finishes so that ACFL knows that the file code can be made available to another file, if required. ASFL also opens files after they have been allocated, closes those files prior to their actual deallocation, and performs password checking for output PSL random library-section files. Assign a File (ASFL), Release a File (RLFL) and Release All Files (RLAF) are the three entry points of module ACFL.

2.2.1.3.1.2.1 ASFL - Assign a File

Module ASFL controls the allocation, deallocation, opening and closing of the files in a PSL library. All PSL modules which access PSL library files call ASFL to allocate and open the file prior to reading or writing it with the PSL access method routines. If file assignment is successful, a file number and for library-section files the section options are returned to the calling program. The file number becomes the unique file identifier used by subsequently called access method routines. If allocation for output is requested for a library-section, the input password is checked against the password stored in the library-section file control block and file assignment is denied if they do not match.

a. Program Operations

HIPO diagram 1.3.1.2.1 describes the program operations. The allocation table is used to keep track of which files have been allocated, which file-codes, and whether the files have been released. When assignment of a file is requested, the allocation table is first searched (step 1) to determine if the file is already allocated. If it is and the access type (input or output) is the same as requested, a new allocation is not required (step 6). For sequential files, in steps 6(1) and 6(2) the file is closed and reopened. If the file is allocated but with a different access type than that requested, the file is deallocated and allocated again to the same file code with the required access type (steps 6(3) through 6(7)). In this way, no file will ever be allocated to more than one file code at the same time. If the file is not currently allocated in step 2, ASFL attempts to find a "free" file code, one to which no file is currently assigned, and in steps 3 through 5 the file is allocated to that file code. If no free file code is found, ASFL finds a "released" file code, one to which a file is currently allocated but which was subsequently released by the module which assigned it. The allocated file is deallocated and the requested file is allocated to that file code.

b. Data File and Table Descriptions

1. INPUT ARGUMENTS

FMS-CATALOG-FILE-STRING

- Catalog file string of file to be allocated.

Diagram ID: 1.3.1.2.1

Name: ASFL - Assign a File

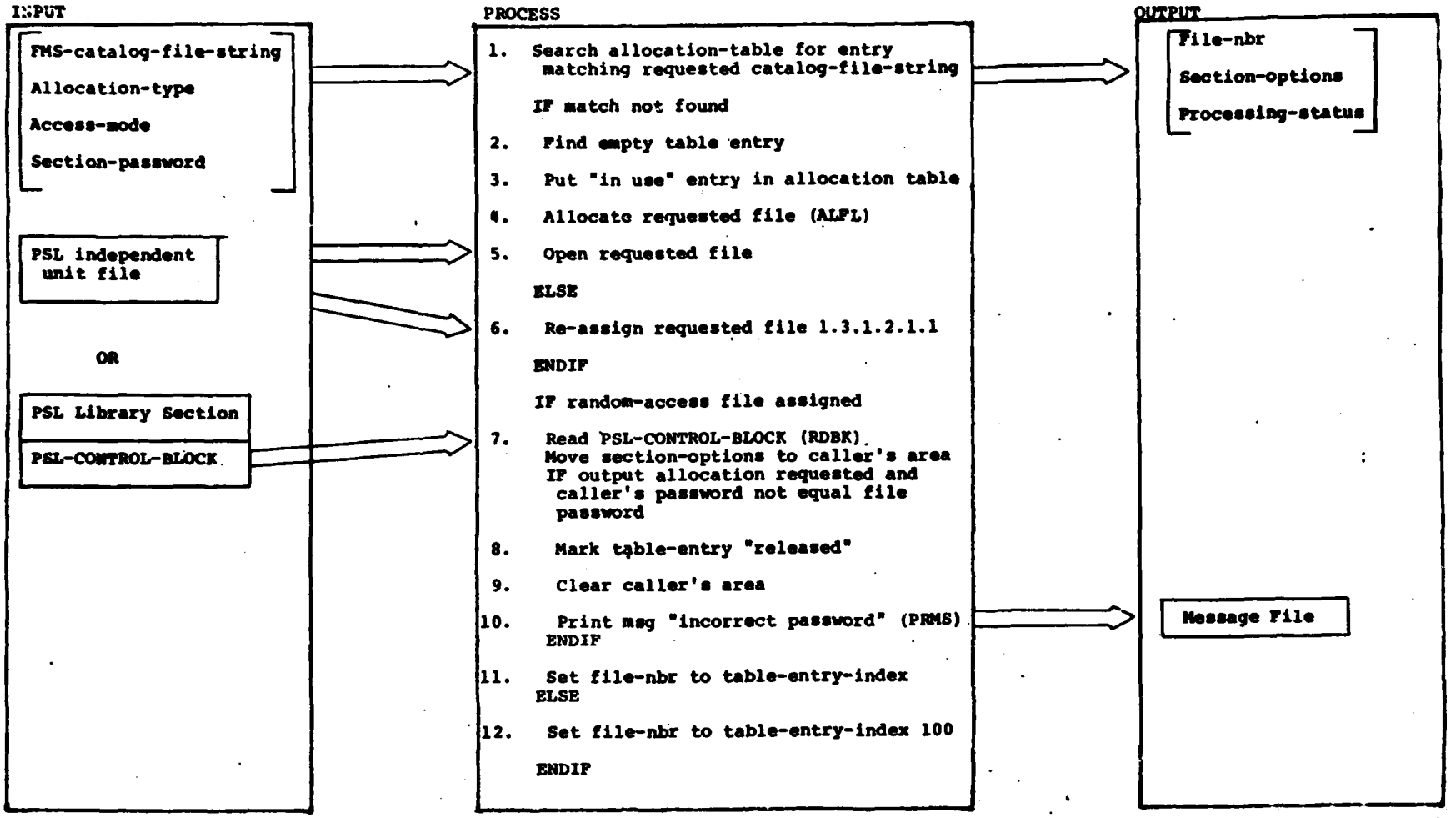
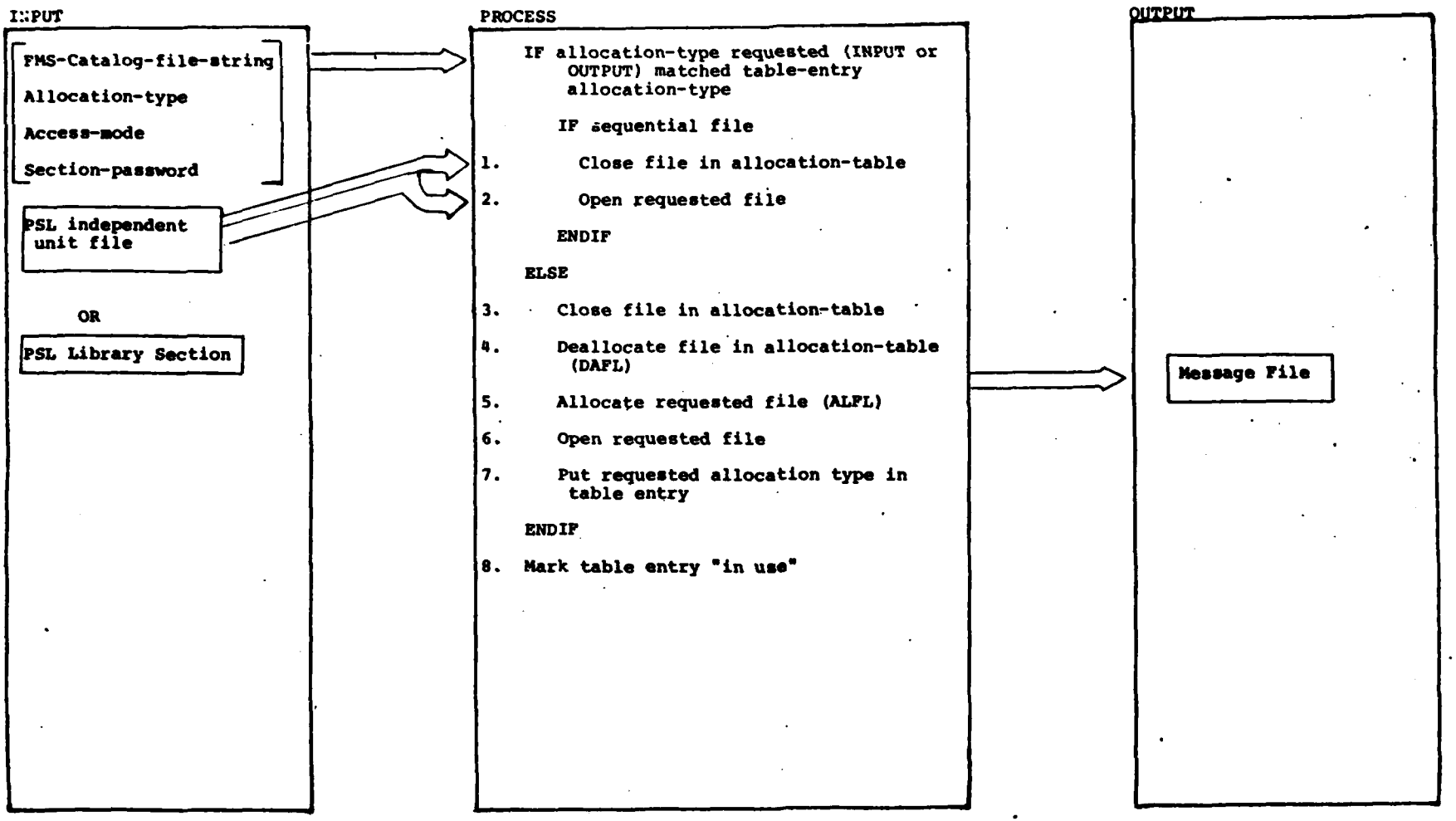


Diagram ID: 1.3.1.2.1

Name: ASFL - Assign a File



- 01 ALLOCATION-TYPE PIC S9(9) COMP.

  Type of allocation requested. Values are:

  1 - Input allocation only.

  2 - Output allocation only.

  3 - Input and output allocation.

  Type of file access to be used. Values are:

  0 - Sequential access.

  1 - Random access.

- 01 SECTION-PASSWORD PIC X(12).

  - Password for random library-section file to be allocated for output. Checked against password stored in control block for library-section file. Not required for input or sequential files.

2. OUTPUT ARGUMENTS

- 01 FILE-NBR PIC S9(9) COMP.

  - Number of file code within the allocation table to which file has been assigned. Sequential files have 100 added to their file number. If file assignment is not successful, the FILE-NBR returned is zero.

- 01 SECTION-OPTIONS.

  - See section 3 for definition of SECTION-OPTIONS. For a random library-section file the SECTION-OPTIONS are read from the control block.

- 01 PROCESSING-STATUS PIC S9(9) COMP.

  Return code - zero for normal status.
  - error code indicating cause of section 2.2.1.3.1.2.1(c) for complete list of codes.

3. ALLOCATION TABLE

The allocation table is the central repository for information concerning the status of file assignments. It is composed of two parts, the RANDOM-ALLOCATION-TABLE for random files and the SEQ-ALLOCATION-TABLE for sequential files.

```
01  RANDOM-ALLOCATION-TABLE-ENTRIES.

    05  RANDOM-ALLOCATION-TABLE OCCURS 4 TIMES

        INDEXED BY R-TABLE-PTR.

        10  RANDOM-FILE-CODE            PIC X(2).
```

- File code associated with this table entry. Values are R1, R2, R3 and R4.

```
        10  FILLER                      PIC X(4).

        10  RANDOM-ENTRY.

            15  RANDOM-FILE-STRING      PIC X(72).
```

- The FMS catalog file string of the file string of the file assigned to this entry.

```
            15  RANDOM-ALLOC-TYPE       PIC S9(9) COMP.
```

- Allocation type used for file assigned to this entry. Values are same as those for input argument ALLOCATION-TYPE.

```
            15  RANDOM-ENTRY-STATUS     PIC S9(9) COMP.
```

- Status of file assigned to this entry. Values are:

  zero - no file is assigned to this entry. Entry is "free".

  1 - file is assigned to this entry and has not yet been released. Entry is "in use".

  2 - file has been assigned to this entry and released, but the file has not yet been deallocated or closed. Entry is "released".

```
• 01  SEQ-ALLOC-TABLE-ENTRIES.

      05  SEQ-ALLOCATION-TABLE OCCURS 3 TIMES
          INDEXED BY S-TABLE-PTR.

          10  SEQ-FILE-CODE              PIC X(2).

              - File code used for this table entry.
                Values are F1 and F2 for input sequential
                files (entries 1 and 2) and F3 for the
                output sequential file (entry 3).

          10  FILLER                     PIC X(4).

          10  SEQ-ENTRY.

              15  SEQ-FILE-STRING        PIC X(72).

                  - The FMS catalog file string of the
                    file assigned to this entry.

              15  SEQ-ALLOC-TYPE         PIC S9(9) COMP.

                  - Allocation type used for file
                    assigned to this entry.  Values are
                    1 for entries 1 and 2, and 2 for
                    entry 3.

              15  SEQ-ENTRY-STATUS       PIC S9(9) COMP.

                  - Status of file assigned to this
                    entry.  Values are the same as
                    RANDOM-ENTRY-STATUS.
```

c. Branching and Error Conditions

Function Reference	Condition Code	Message Category	Program Action	Note
2	6	ADV	File is not assigned because no file codes are available.	
4,6(5)	2	FMS	File not assigned.	1
7	33	PSL	File not assigned.	2
8,10	97	ERR	File not assigned.	

Notes:

(1) Unable to allocate requested file. See Sperry Univac 1100 series Executive System Volume 2 Exec Appendix C section C.2 Facility Request Status Codes for messages returned.

(2) Unable to read a block of the PSL library-section file. See description of ACBK for specific causes of "unable to read a block" condition.

2.2.1.3.1.2.2 RLFL - Release a File

Module RLFL changes the status of an assigned file in the file allocation table from "in use" to "released". The "released" status indicates to subsequent calls to ASFL that the file may be deallocated, if required, to re-use the file code for the allocation of another file. When a PSL module uses ASFL to allocate a PSL file, the module is expected to call RLFL to release the file when it is finished processing that file.

a. Program Operations

HIPO diagram 1.3.1.2.2 describes the program operations. See descriptions of ACFL and ASFL for detailed description of use of allocation table and its entries.

b. Data File and Table Descriptions

1. INPUT ARGUMENTS

01 REQUESTED-FILE-NBR PIC S9(9) COMP.

- Number of file to be released. This is the number assigned to the file when it was assigned by ASFL.

2. OUTPUT ARGUMENTS

01 PROCESSING-STATUS PIC S9(9) COMP.

Return code - zero for normal return.
- 22 for invalid file number requested.
- 37 for release file error.

3. TABLE DESCRIPTIONS

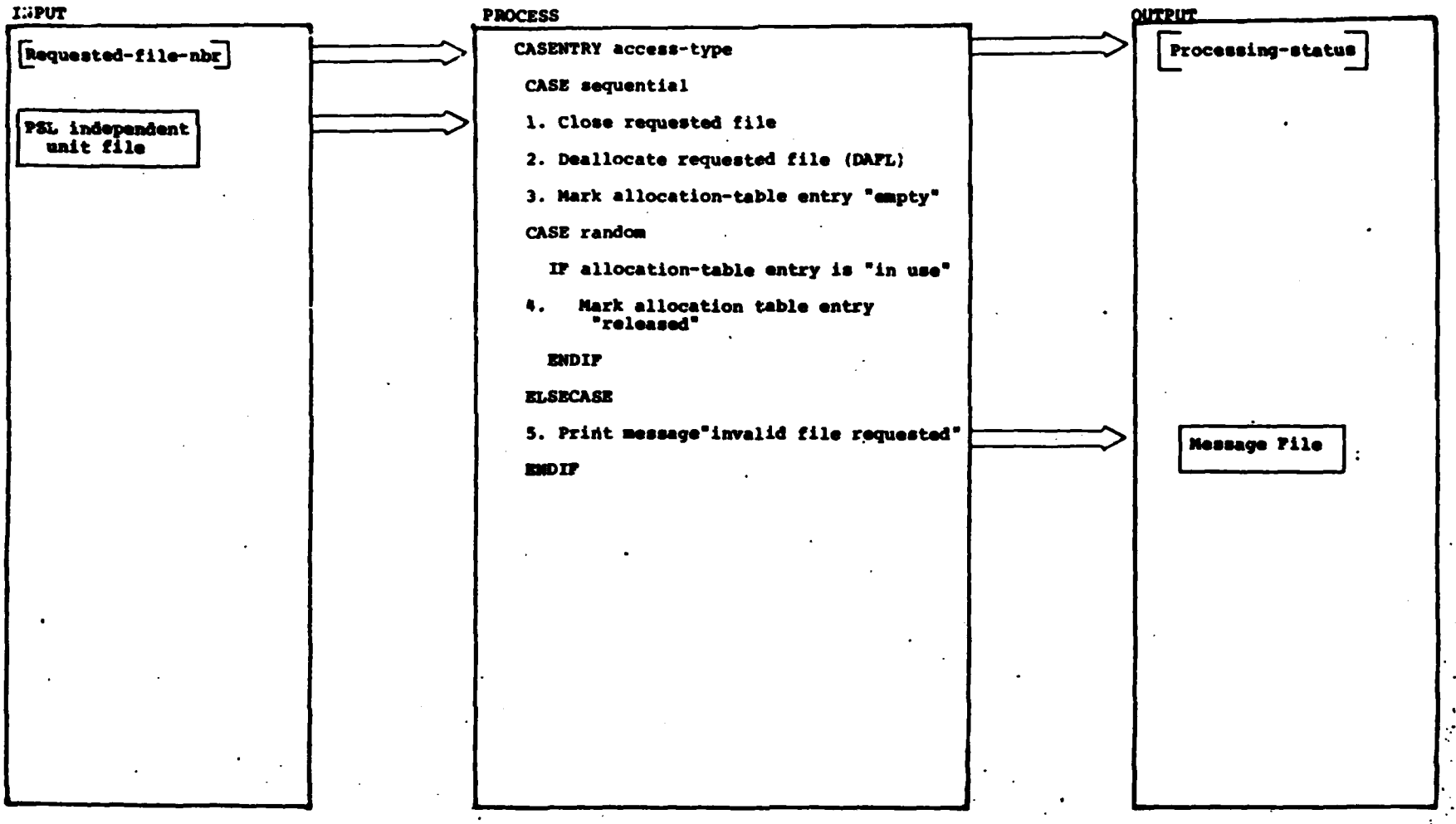
The allocation table is described with module ASFL.

c. Branching and Error Conditions

Function Reference	Condition Code	Message Category	Program Action	Note
2	37	PSL	File is released. Error return from DAFL may have unpredictable result.	
5	22	PSL	File is not released.	

Diagram ID: 1.3.1.2.2

Name: RLFL - Release a File



2.2.1.3.1.2.3 RLAF - Release all Files

Module RLAF closes and deallocates all files which have been previously assigned by ASFL but not yet deallocated.

a. Program Operations

HIPO diagram 1.3.1.2.3 describes the program operations. See description of ASFL for definition of file statuses, "in use", "released" and "empty".

b. Data File and Table Descriptions

1. OUTPUT ARGUMENT

01 PROCESSING-STATUS PIC S9(9) COMP.

Return code - zero for normal processing.
- 37 for release file error.

2. TABLE DESCRIPTIONS

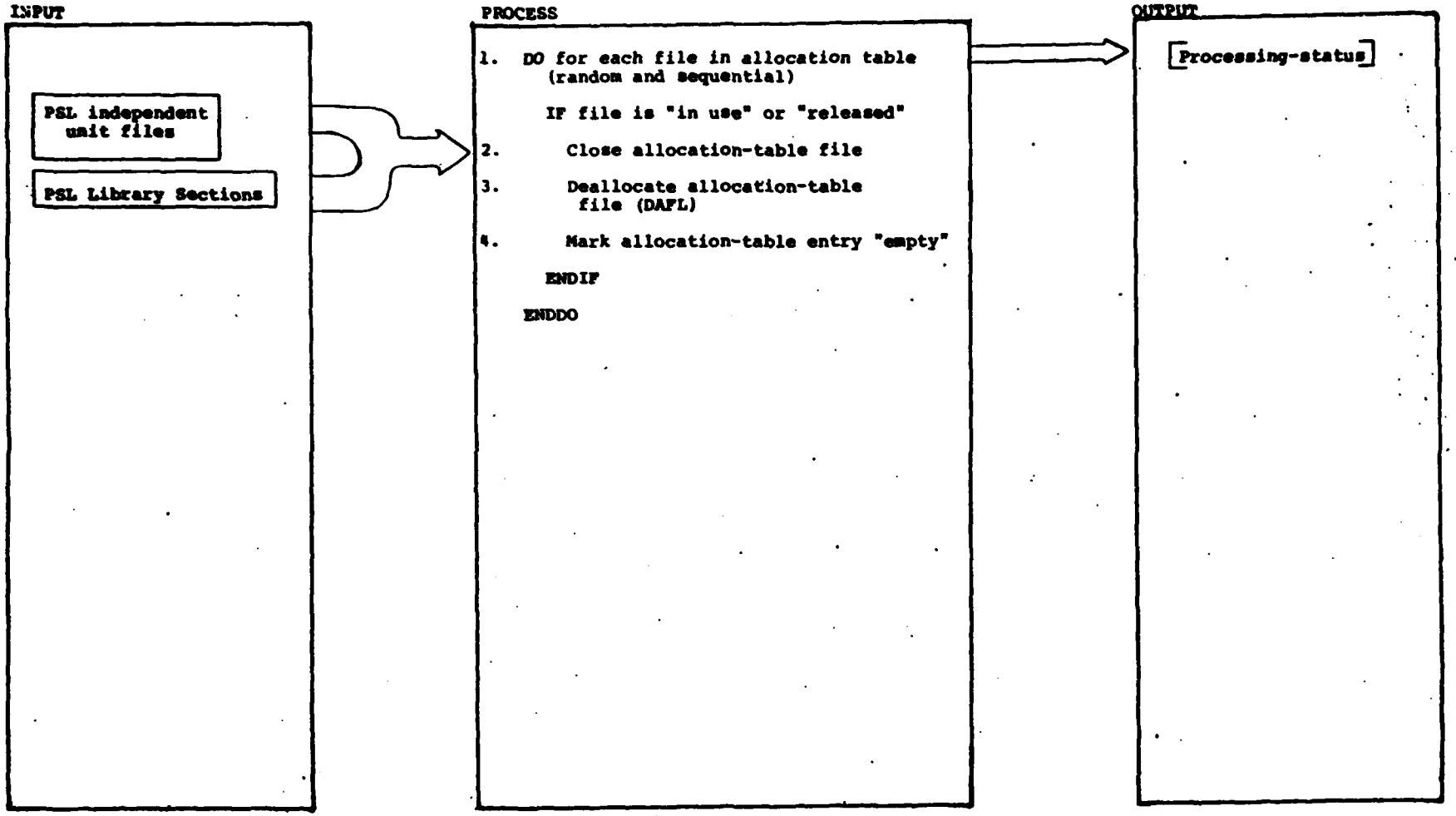
The allocation table is described with module ASFL.

c. Branching and Error Conditions

Function Reference	Condition Code	Message Category	Program Action	Note
2	37	PSL	File is released. Error return from DAFL may have unpredictable result.	
5	22	PSL	File is released.	

Diagram ID: 1.3.1.2.3

Name: RLAF - Release all Files

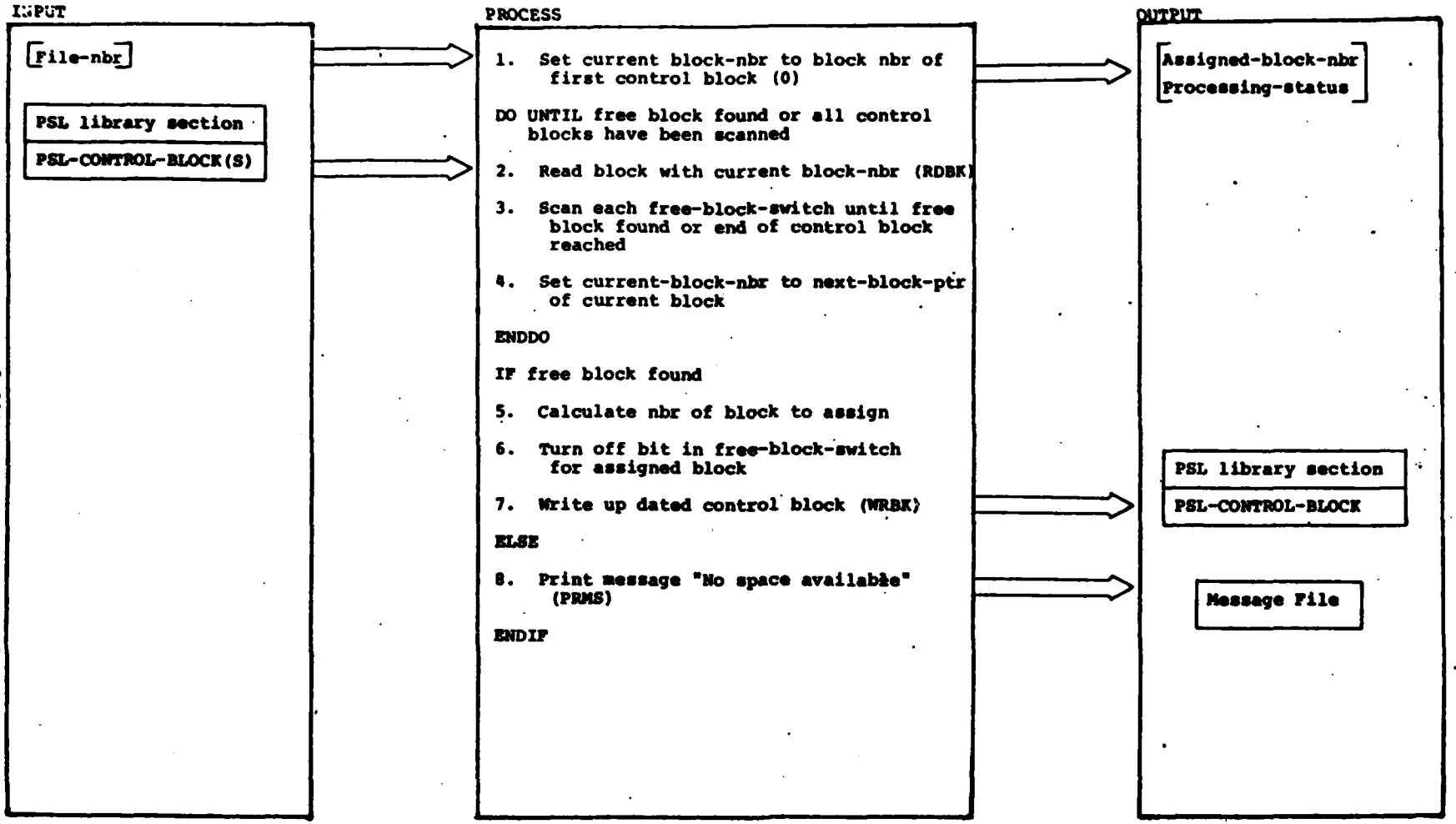


2.2.1.3.1.3 ACBK - Access a Block

The ACBK module through its four entry points does almost all of the actual input and output (I/O) on the PSL library-section files as well as space management within the files. The entry points are: Assign a Block (ASBK), Release a Block (RLBK), Read a Block (RDBK), and Write a Block (WRBK). To the ACBK routine the PSL library-section file is handled as a group of 128-word blocks which are assigned to a calling program as needed and released to the pool of free blocks at the request of the caller when the block is no longer needed. The first block of the file, called the control block, is reserved for this space management function. It contains a series of free block switches whose value indicate whether or not a particular block of the file is in use. Each block of the file except the first (block zero) is represented by one bit of the free block switches. If there are not enough switches in the first control block to represent all blocks of the file, additional blocks will be assigned to contain additional switches. Subsequent blocks are chained to the first with a pointer in the second word of the block. A value of one (1) indicates that the corresponding block is "free"; a value of zero (0) indicates that the corresponding block is "in use". Each character position of the free block switches represents six blocks in the PSL library-section file. Appropriate calculations are done, all in COBOL, to turn on and off the appropriate bits for each block.

Each block of the PSL library-section file is formatted such that the first word of the block contains its relative block number within the random file. The second word is reserved for use as a chain pointer so that if the block is chained to another block, the second word will contain the relative block number of the preceeding block in the chain. If the block is not chained to another block, the second word will always contain zero. The ACBK routines assume that the PSL library-section file has been initialized in this manner, probably with a call to Initialize a Standard File (ITSF) in a prior run, and that the PSL library-section file has been allocated and opened by a previous call to Assign a File (ASFL) in the same run.

Diagram ID: 1.3.1.3.1 Name: ASBK - Assign a Block



2.2.1.3.1.3.1 ASBK - Assign a Block

The module ASBK locates a free block in a PSL library-section file and returns the block number to the calling program.

a. Program Operations

HIPO diagram 1.3.1.3.1 describes the program operations.

b. Data File and Table Descriptions

1. INPUT ARGUMENTS

- 01 FILE-NBR PIC S9(9) COMP.

  - number assigned to library-section file when it was allocated by ASFL.

2. OUTPUT ARGUMENTS

- 01 ASSIGNED-BLOCK-NBR PIC S9(9) COMP.

  - relative block number of block within the PSL library-section file which has been assigned for use by the calling program.

- 01 PROCESSING-STATUS PIC S9(9) COMP.

  Return code - zero for normal processing.
  - 41 for unable to assign a block.

c. Branching and Error Conditions

Function Reference	Condition Code	Message Category	Program Action	Note
2,7	22	PSL	Block not assigned.	
2,7	94	PSL	Block not assigned.	
2	33	PSL	Block not assigned.	
7	34	PSL	Block not assigned.	
8	5	ERR	Block not assigned.	
Any	41	PSL	Previous error prevented block from being assigned.	

2.2.1.3.1.3.2 RDBK - Read a Block

The module RDBK reads a block with the given relative block number from a PSL library-section file.

a. Program Operations

HIPO diagram 1.2.1.3.2 describes the program operations.

b. Data File and Table Descriptions

1. INPUT ARGUMENTS

- 01 FILE-NBR PIC S9(9) COMP.

  - number assigned to library-section file when it was allocated by ASFL.

- 01 BLOCK-NBR PIC S9(9) COMP.

  - relative block number of block to be read.

2. OUTPUT ARGUMENTS

- 01 RANDOM-RECORD-AREA PIC X(768).

  - record (block) read from PSL library-section file. See section 3 for description of formats of various record types in PSL library-section file.

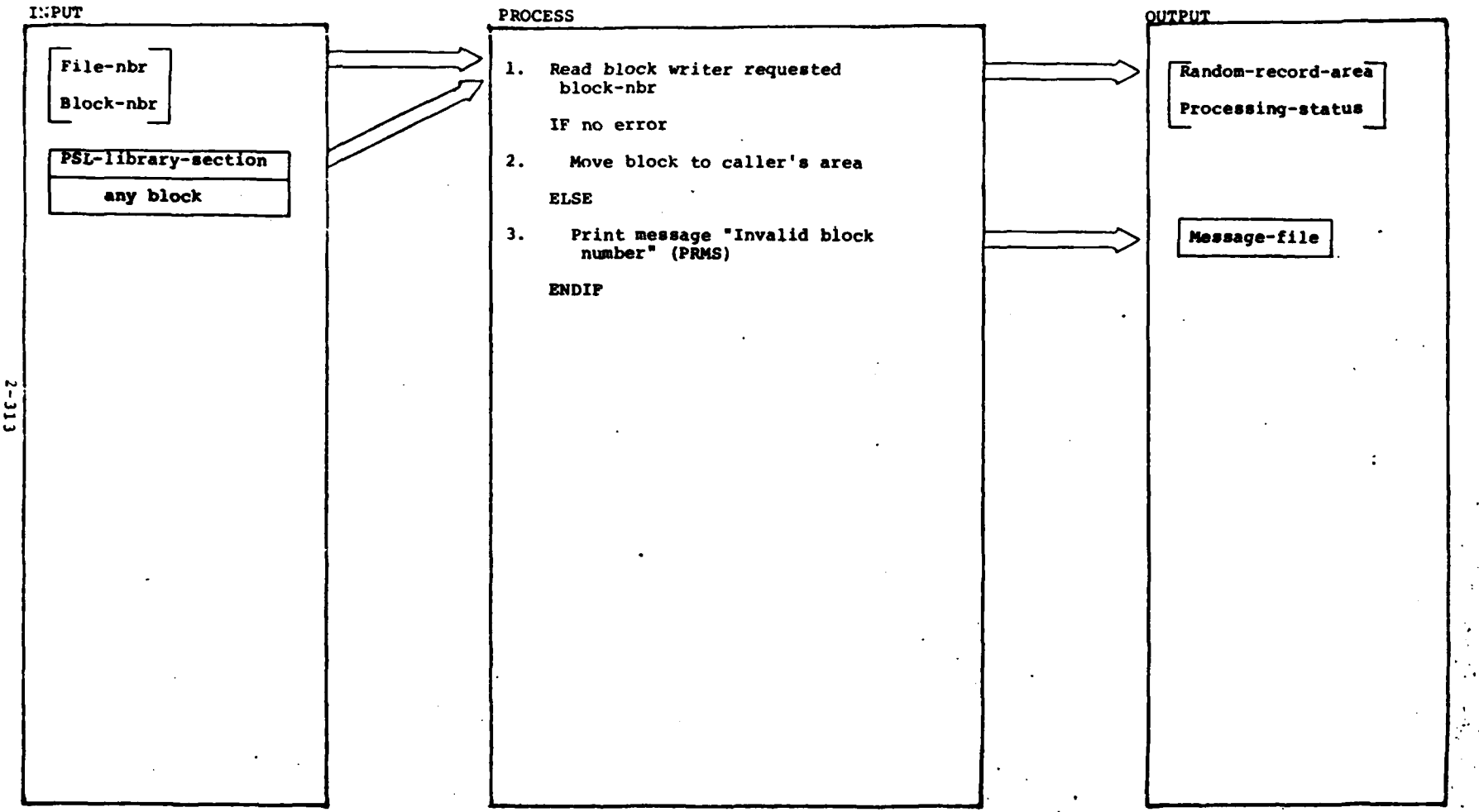
- 01 PROCESSING-STATUS PIC S9(9) COMP.

  Return code - zero for normal processing.
  - 33 for unable to read a block.

c. Branching and Error Conditions

Function Reference	Condition Code	Message Category	Program Action	Note
1	22	PSL	Block is not read.	
1	94	PSL	Block is not read.	
Any	33	PSL	Previous error prevented block from being read.	

Diagram ID: 1.3.1.3.2 Name: RDBK - Read a Block



2.2.1.3.1.3.3 WRBK - Write a Block

The module WRBK writes a block with a given relative block number to a PSL library-section file.

a. Program Operations

HIPO diagram 1.3.1.3.3 describes the program operations. It is required that the first word of the blocks of the PSL library-section file contain the relative block number of the block. If the block provided by the calling program does not meet this requirement, the block will not be written to the file.

b. Data File and Table Descriptions

1. INPUT ARGUMENTS

- 01 FILE-NBR PIC S9(9) COMP.

  - number assigned to library-section file when it was allocated by ASFL.

- 01 BLOCK-NBR PIC S9(9) COMP.

  - relative block number of block to be written.

- 01 RANDOM-RECORD-AREA PIC X(768).

  - completed block to be written to PSL library-section file. See section 3 for description of formats of various record types in PSL library-section file. Block is written to file exactly as input. If another block is to be chained to this one, the calling program must have put the relative block number of that block in the second word prior to calling WRBK.

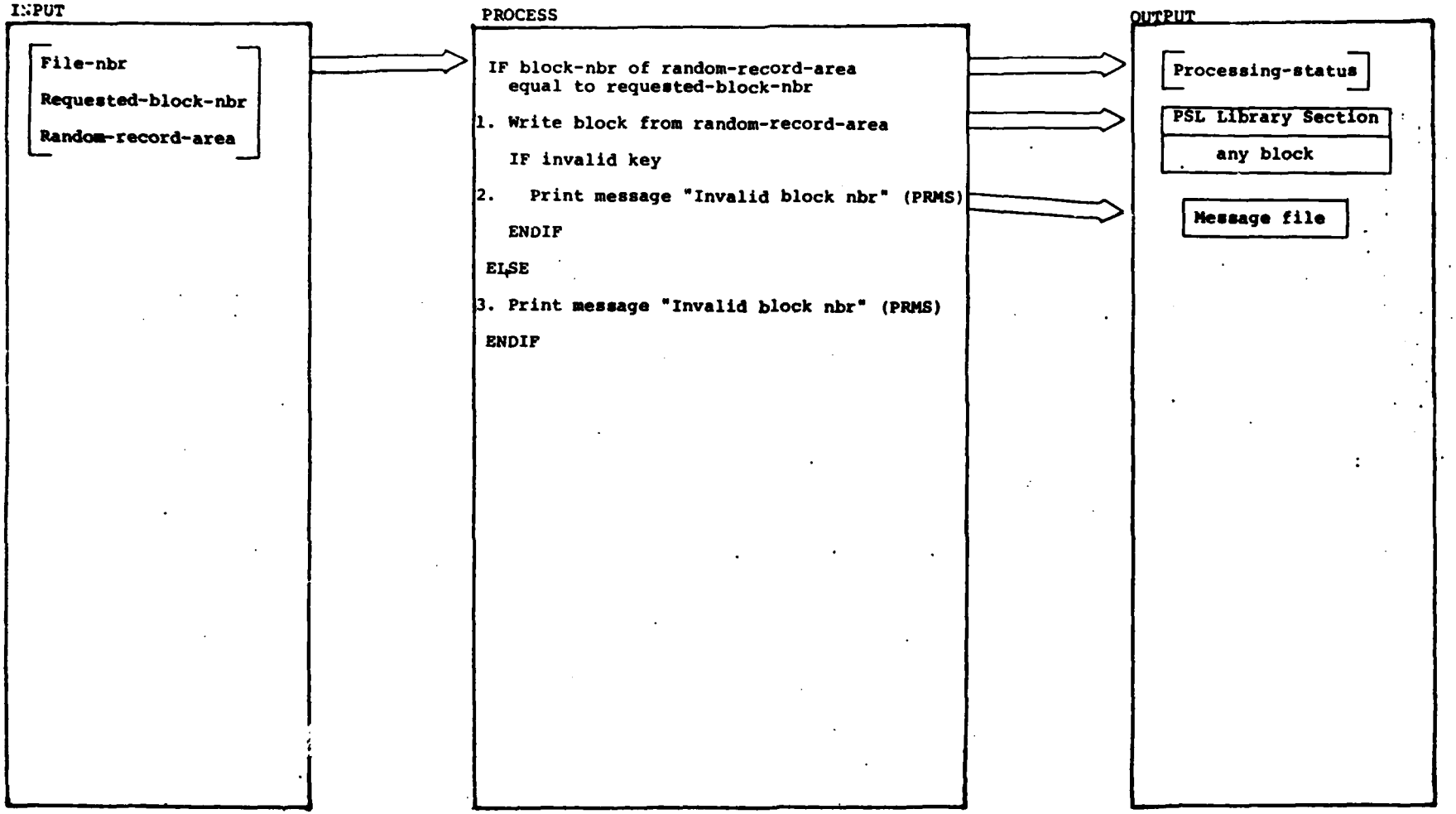
2. OUTPUT ARGUMENTS

- 01 PROCESSING-STATUS PIC S9(9) COMP.

  Return code - zero for normal processing.
  - 34 for unable to write a block.

Diagram ID: 1.3.1.3.3

Name: WRBK - Write a Block



c. Branching and Error Conditions

Function Reference	Condition Code	Message Category	Program Action	Note
1	22	PSL	Block is not written.	
1	94	PSL	Block is not written.	
Any	34	PSL	Previous error prevented block from being written.	

2.2.1.3.1.3.4 RLBK - Release a Block

The module RLBK releases a block of a PSL library-section file to make it available for re-use as required. When other blocks are chained to the specified block, all blocks in the chain are released.

a. Program Operations

HIPO diagram 1.3.1.3.4 describes the program operations. In step 5, turning on the bit in the free block switches which corresponds to the requested block makes that block available for re-assignment by ASBK. In steps 7 through 8, the released block is read and word 2 of the block is tested to determine if another block is chained to it and thus needs to be released. Each released block is zeroed and re-written.

b. Data File and Table Descriptions

1. INPUT ARGUMENTS

01 FILE-NBR PIC S9(9) COMP.

- number assigned to library-section file when it was allocated by ASFL.

01 BLOCK-TO-BE-RELEASED PIC S9(9) COMP.

- relative block number of block to be released.

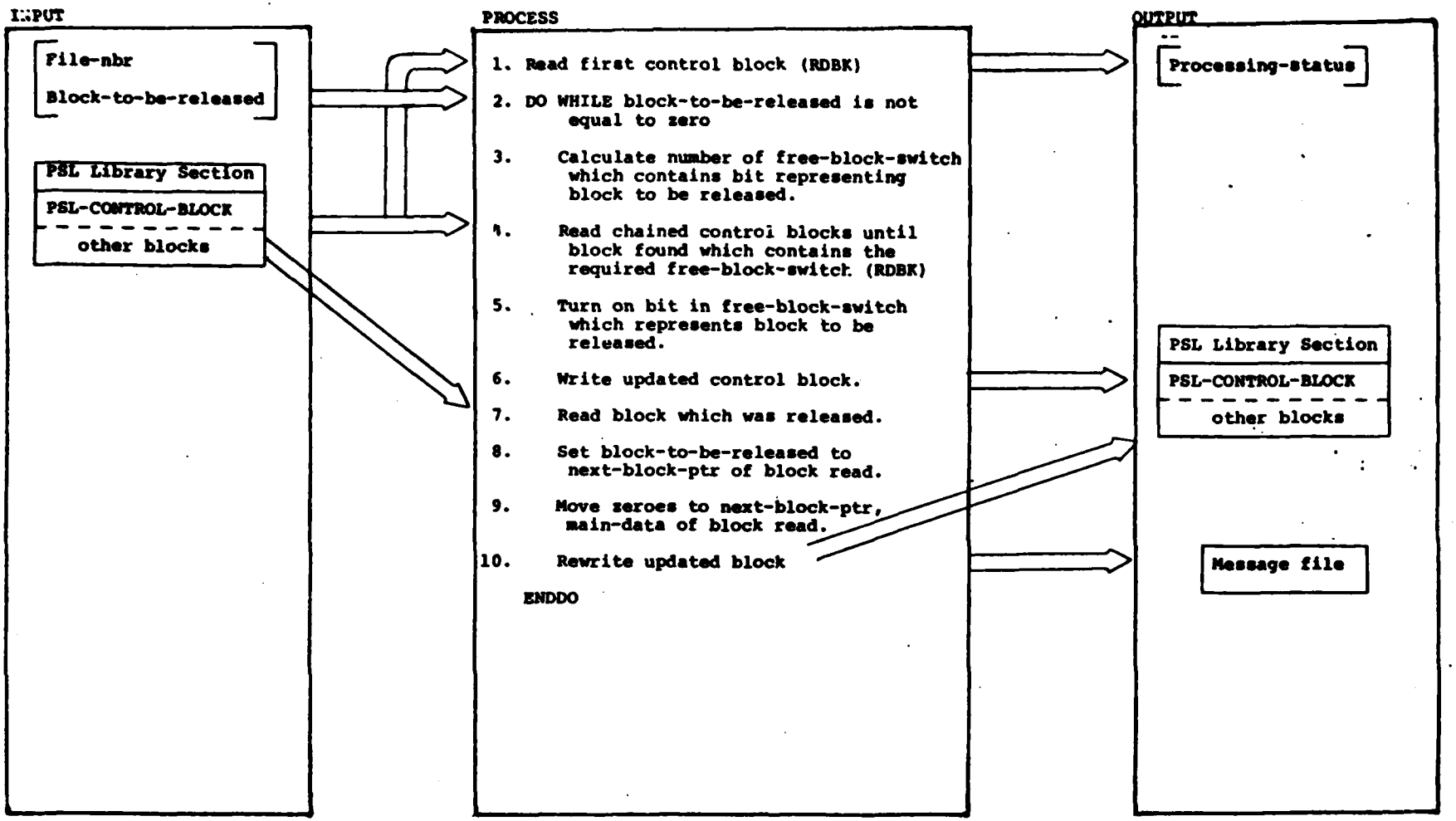
2. OUTPUT ARGUMENTS

01 PROCESSING-STATUS PIC S9(9) COMP.

Return code - zero for normal processing.
- 42 for unable to release a block.

Diagram ID: 1.3.1.3.4

Name: RLBK - Release a Block



c. Branching and Error Conditions

Function Reference	Condition Code	Message Category	Program Action	Note
1,4	33	PSL	Block is not released.	
4	12	PSL	Subsequent processing is bypassed.	
6	34	PSL	Block is not released.	
10	34	PSL	Block is released; block is not zeroed; chained blocks are not released.	
7	33	PSL	Block is released; block is not zeroed; chained blocks are not released.	

2.2.1.3.2 Unit I/O

Unit I/O is performed to read and write the data elements that constitute a unit in section file storage. The program modules which perform these operations are described below.

2.2.1.3.2.1 WRAC - Write Accounting Information

The WRAC module is called to modify the information in the unit accounting information segment of a unit data block.

a. Program Operations

HIPO diagram 1.3.2.1 depicts the program operations performed by the WRAC module. If the calling program module does not indicate the block number of the PSL data block to be modified (i.e., REQUESTED-DATA-BLOCK-NBR equals zero), then the requested unit name is used to find the given unit in the section index. If the unit is not found then it is presumed that a data block needs to be assigned and initialized. The block number of the given unit is thus determined in one of three ways: 1) it is given, 2) it is found, or 3) it is assigned. In all cases the determined data block is modified with the given accounting information and the PSL data block is written to the section file. In the latter case, it remains for the calling program to add the unit name to the section index following a normal return from the WRAC module.

b. Data File and Table Descriptions

No special files or tables are utilized by the WRAC module.

c. Branching and Error Conditions

The following branching and error conditions apply to HIPO diagram 1.3.2.1:

Function Reference	Condition Code	Message Category	Program Action	Note
7	46	PSL	Exit module	1

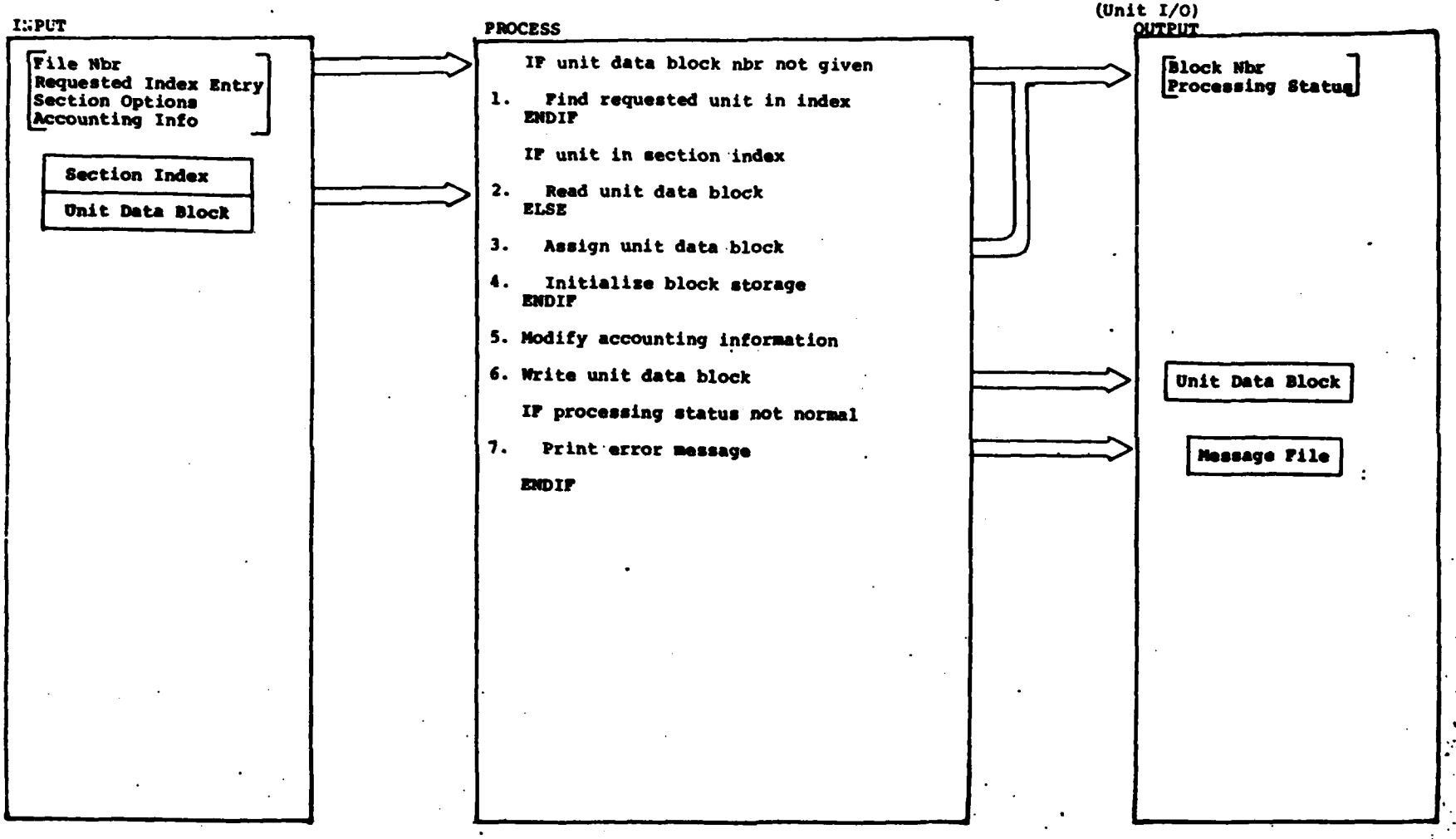
NOTES:

(1) This error may occur as a secondary error when preceded by an error return from the ASBK module relative to a "no-space-available" error message (condition code = 5); otherwise it is unlikely to occur.

Diagram ID: 1.3.2.1

Name: WRAC - Write Accounting Information

Description: PSL Access Routine (Unit I/O)



2.2.1.3.2.2 RDUN - Read a Unit

The module RDUN is used by all PSL function processors to read the accounting record and data lines of a unit in a PSL library section file. RDUN consists of three entry points, Initialize to Read (ITRD), Read a Line (RDLN), and Terminate Read (TMRD). The calling program must call ITRD to initalize the reading of a specific unit. A successful return from ITRD is then followed by repetitive calls to RDLN to obtain in sequence, the lines of data contained in the unit. A call to TMRD is required only if the calling program does not continue to call RDLN until an end-of-unit return is received. For PSL units which are not typed INDEPENDENT, that is whose contents are completely contained in the library-section random file, unit-reads may be nested. If a second call to ITRD is made before the previous unit read is complete, that is neither end-of-unit has been detected nor a call to TMRD has been made, the current status of the unit-read in progress is saved in a push down stack, and subsequent calls to RDLN will return, in sequence, the lines of the newly initialized unit. Occurrence of an end-of-unit condition or a call to TMRD will discontinue reading of the new unit and restore the read status of the most recently saved unit-read. Subsequent calls to RDLN then continue to return the lines of the interrupted unit as though no interruption had occurred. Unit-reads may be nested in this way up to a maximum depth of 50. INDEPENDENT units can never be nested, either with other INDEPENDENT units or with random units. An attempt to do so will result in an error return from ITRD.

2.2.1.3.2.2.1 ITRD - Initialize to Read

The module ITRD initializes data areas and buffers required to prepare for reading the lines of data of a unit in a PSL library-section. The accounting record of the unit to be read is returned to the caller.

a. Program Operations

HIPO diagram 1.3.2.2.1 describes the operations performed. It is assumed that the library-section file from which the unit is to be read has been allocated by a call to ASFL prior to calling ITRD. If the reading of another random unit is in progress, the status of that unit-read is saved in push-down stack STACK-OF-ACTIVE-UNITS (step 1). If the number of the first data block of the unit is not present (is equal to zero) in the input index entry, ITRD will obtain a complete index entry by calling FDXE (step 2). For the initialization of a read for an INDEPENDENT unit, the input FMS catalog file string is passed to ASFL to allocate and open the independent unit file prior to its being read (step 5). In step 6, if the data in the section is compressed, the position of the first character of the first line of the unit is computed to be the first character position after the accounting information or extended accounting information if the MGMT option is in effect for the section. If the data in the section is not compressed, the pointer to the first line of the unit is set to skip the number of lines of space which is occupied by the accounting information and the extended accounting information if the MGMT option is in effect for the section.

b. Data File and Table Descriptions

1. INPUT ARGUMENTS

- 01 IN-FILE-NBR PIC S9(9) COMP.

  - number assigned to library-section file when it was allocated by ASFL.

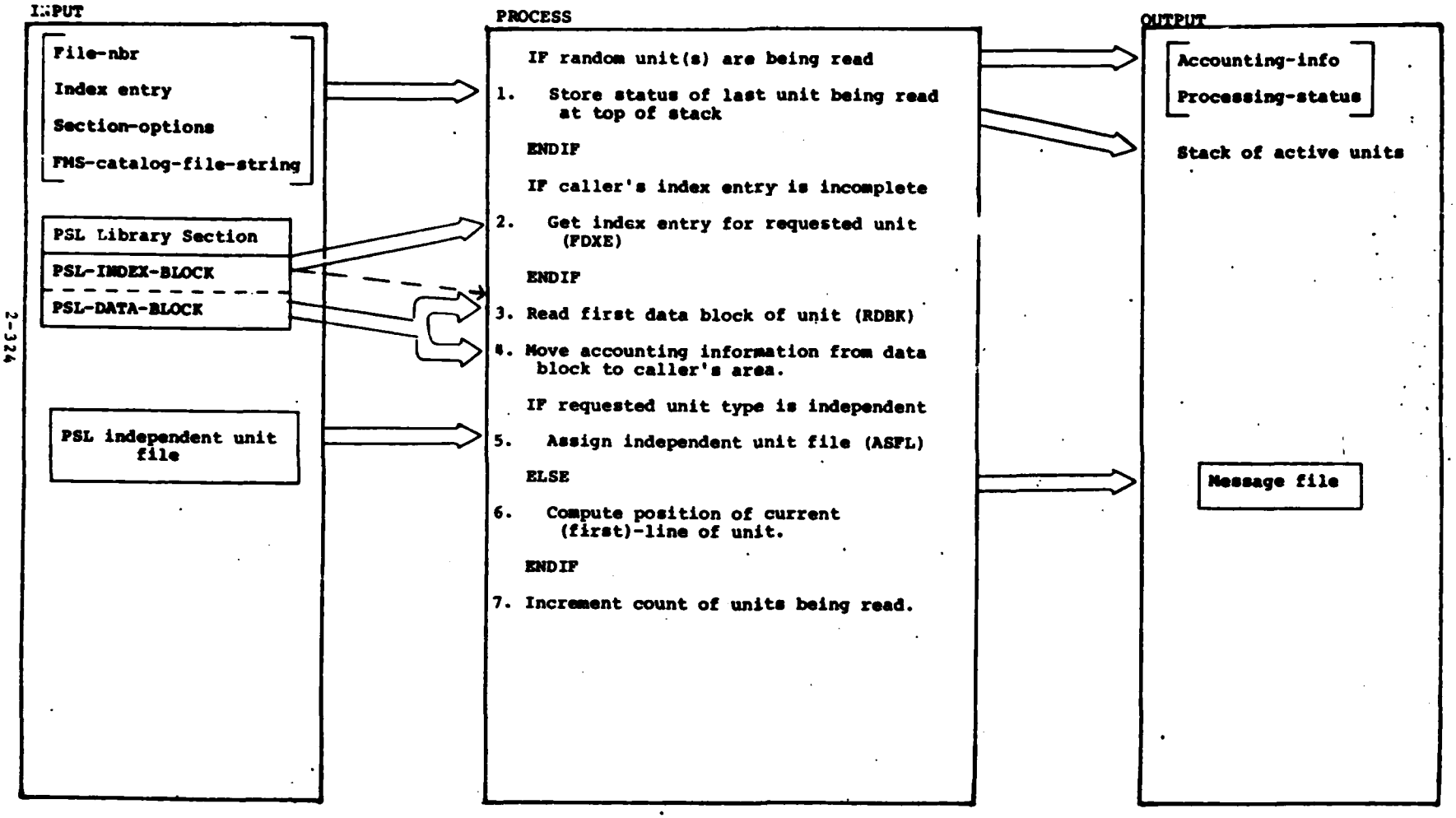
- 01 REQUESTED-INDEX-ENTRY.

  05 REQUESTED-UNIT-NAME PIC S9(9) COMP.

  - name of unit to be read.

Diagram ID: 1.3.2.2.1

Name: ITRD - Initialize to Read



05 REQUESTED-DATA-BLOCK-PTR PIC S9(9) COMP.

- number of first data block of unit to be read. If this field contains zero, a new index entry will be obtained by ITRD.

05 REQ-INCLUDING-UNITS-LIST PIC S9(9) COMP.

- Not used.

05 REQUESTED-UNIT-TYPE PIC X(1).

- unit type code for unit to be read. Code must be one of those defined in the UNIT-TYPE-TABLE.

- 01 IN-SECTION-OPTIONS PIC X(16).

- SECTION-OPTIONS field received from ASFL when library-section to be read was allocated.

- 01 FMS-CATALOG-FILE-STRING.

- FMS catalog file string as defined in section for independent unit file when unit typed INDEPENDENT is to be read. Not used for random units.

2. OUTPUT ARGUMENTS

- 01 ACCOUNTING-INFO.

- Accounting information read from first data block of unit to be read. Fields of accounting information area are described in section .

- 01 PROCESSING-STATUS PIC S9(9) COMP.

Return code - zero for normal processing.
- 47 for unable to initialize to read.

3. STACK-OF-ACTIVE-UNITS

This push-down stack contains the status of those units whose reading was interrupted when another unit was initialized for reading. Each entry in the stack contains all information for one interrupted unit.

05 STACK-ENTRY OCCURS 50 TIMES.

10 STACK-FILE-NBR PIC S9(9) COMP.

- File number of file from which unit was being read.

10 STACK-BLOCK-NBR PIC S9(9) COMP.

- Block number of last block of unit which was being read.

10 STACK-LINE-COUNTER PIC S9(9) COMP.

- number of lines read within the current data block.

10 STACK-LINE-POINTER PIC S9(9) COMP.

- for compressed data, pointer to first character of next data line to read. For non-compressed data, index to position of last data line read. Not used for independent units.

10 STACK-COMPRESSED-SW PIC 9.

- Flag to indicate whether the library-section file being read has its data compressed. Value zero is for not compressed; value one is for compressed. Not used for INDEPENDENT units.

2.2.1.3.2.2.2 RDLN - Read a Line

The module RDLN obtains the next data line in sequence of a unit in a PSL library-section file. Units which can be read are restricted to card-image data in the SOURCE, JOB, LINK, TEST, TEXT, MGMT and PDL sections.

a. Program Operations

HIPO diagram 1.1.2.2.2 describes program operations. For random units all data lines are stored in the library-section file itself. Data blocks are chained together and are read in sequence as required. In the data area of each data block, the data records are stored in sequence. For non-compressed data, the data area consists of a sequence of 80 character card-image records. For compressed data, the data records are variable length. The first two bytes (12 bits) of each record contains the binary byte number within the data area of the first byte of the next data record. The second two bytes of each data record contains, in binary, the number of leading blank characters contained in the non-compressed data line stored in the record. These two control fields are followed by the actual characters of the data line. Leading and trailing blanks which occurred in the original non-compressed line are not stored, but are inserted by RDLN in the data line which is returned to the caller. For units typed INDEPENDENT, the data records are stored in a separate file from the library-section file. This independent file is in system standard format with 80-column card-image records. Data records are obtained using the standard COBOL sequential read.

b. Data File and Table Descriptions

1. INPUT DATA ITEMS

The data items maintained from one call of RDLN to another are the same as the unit-read status items stored in the STACK-OF-ACTIVE-UNITS described in section 2.2.1.3.2.2.1. The values for the unit currently being read are stored in the following data items:

- 01 FILE-NBR PIC S9(9) COMP.

  - file number of file from which unit was being read.

- 01 BLOCK-NBR PIC S9(9) COMP.

  - block number of last block of unit which was being read.

Diagram ID: 1.3.2.2.2

Name: RDLN - Read a Line

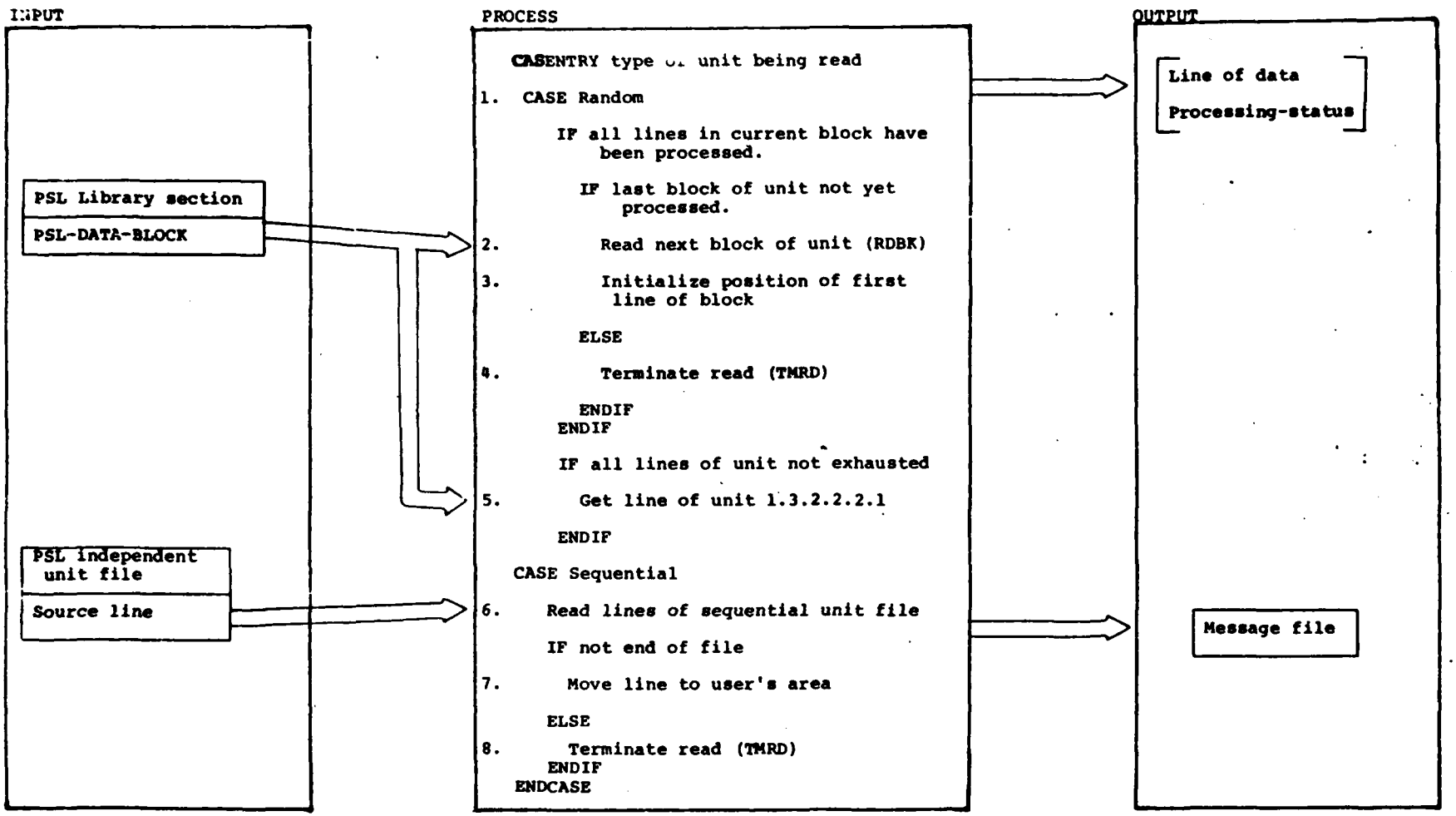
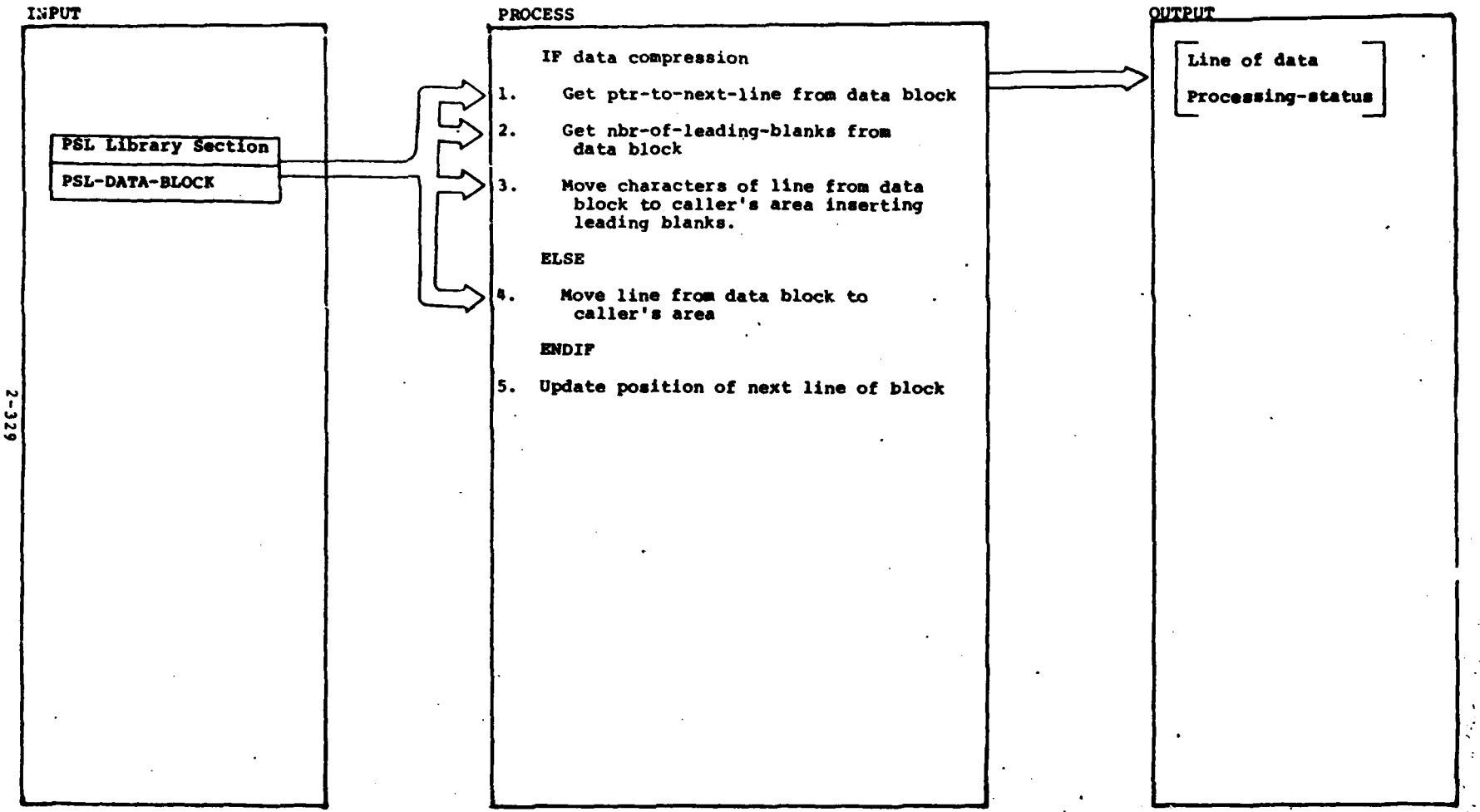


Diagram ID: 1.3.2.2.2.1

Name: RDLN - Get Line of Unit



- 01 LINE-COUNTER PIC S9(9) COMP.

  - number of lines read within the current data block.

- 01 PTR-TO-NEXT-LINE PIC S9(9) COMP.

  - for compressed data, pointer to first character of next data line to read.

- 01 LINE-PTR PIC S9(9) COMP.

  - for non-compressed data, index to position of last data line read.

- 01 DATA-COMPRESSION-SW PIC 9.

  - Flag to indicate whether the library-section file being read has its data compressed. Value zero is for not compressed; value one is for compressed. Not used for INDEPENDENT units.

2. OUTPUT ARGUMENTS

- 01 LINE-OF-DATA PIC X(80).

  - Data line read from unit. Not returned when end-of-unit is detected.

- 01 PROCESSING-STATUS PIC S9(9) COMP.

  Return code - zero for line of data returned.
  - 18 for end-of-unit
  - 48 for read error.

c. Branching and Error Conditions

Function Reference	Condition Code	Message Category	Program Action	Note
2	33	PSL	Subsequent processing is bypassed.	1
6	22	PSL	Subsequent processing is bypassed.	

c. Branching and Error Conditions (Continued)

5,6	18	N/A	End-of-unit code returned to caller.
Any	48	PSL	Previous error prevented successful unit read. Code returned to caller.

Note:

(1) Unable to read a block from library-section file. See description of module ACBK for conditions which cause "unable to read block" condition.

2.2.1.3.2.2.3 TMRD - Terminate Read

The module TMRD ends the read processing for a unit of a PSL library-section file. If the reading of another unit was interrupted to begin reading the current unit, the status of the previous unit is restored so that reading of that unit can resume.

a. Program Operations

HIPO diagram 1.3.2.2.3 describes the program operations. Current data items to be restored are taken from the STACK-OF-ACTIVE-UNITS.

b. Data File and Table Descriptions

All significant data is described with modules ITRD and RDLN, sections 2.2.1.3.2.2.1 and 2.2.1.3.2.2.1, respectively.

c. Branching and Error Conditions

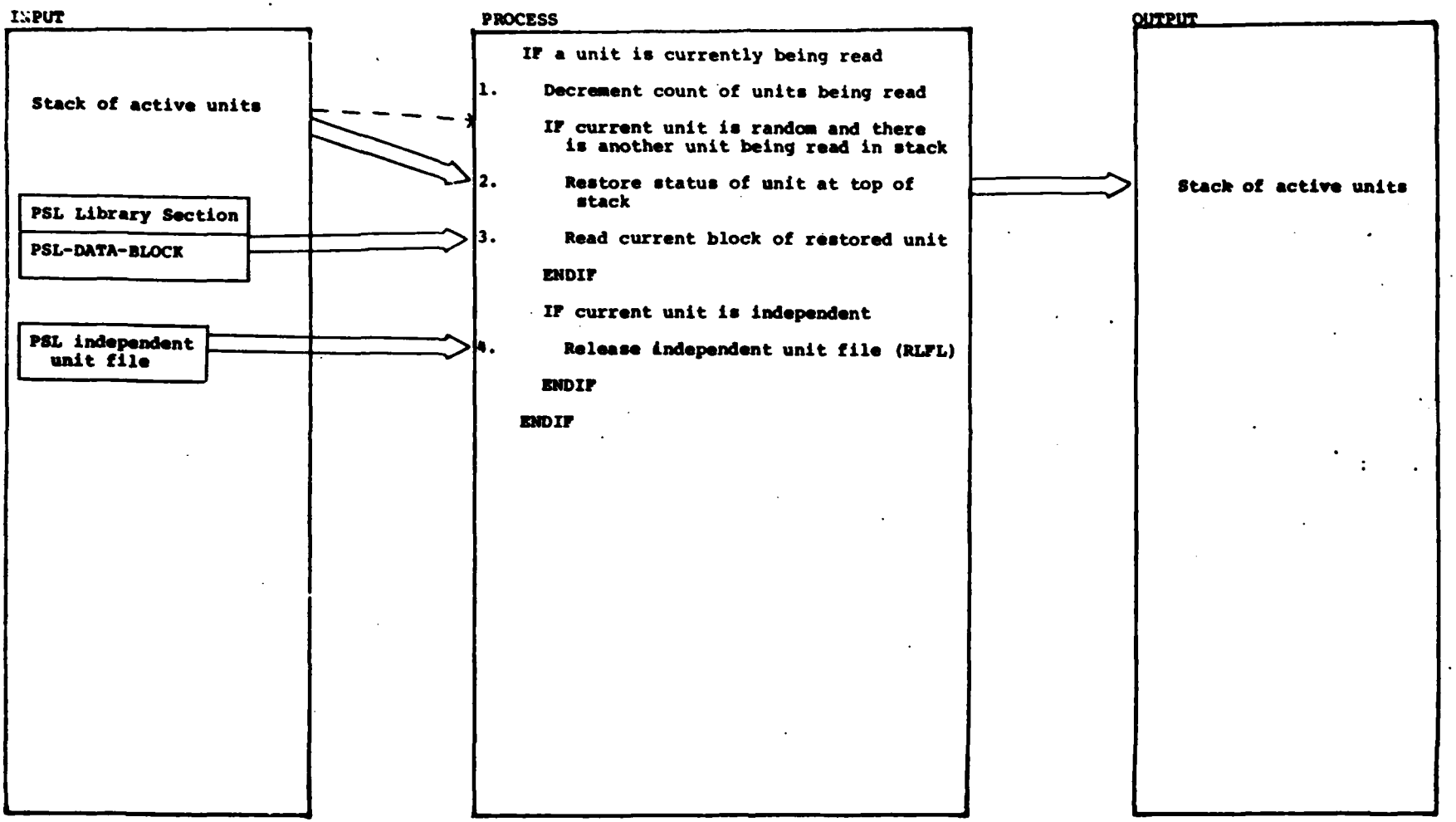
Function Reference	Condition Code	Message Category	Program Action	Note
3	33	PSL	Code 49 returned to caller.	1
3	49	PSL	Code returned to caller due to previous error.	

Note:

(1) Unable to read a block from library-section file. See description of module ACBK for conditions which cause "unable to read block" condition.

Diagram ID: 1.3.2.2.3

Name: TMRD - Terminate Read



2.2.1.3.2.3 WRUN - Write a Unit

The module WRUN is used by all PSL function processors to write the accounting record and data lines of a unit in a PSL library-section file. WRUN consists of three entry points, Initialize to Write (ITWR), Write a Line (WRLN), and Terminate Write (TMWR). WRUN will either write over the data blocks of an existing unit or obtain free data blocks to write the new unit. It is assumed that the library-section file has been allocated with a call to ASFL prior to calling WRUN. The calling program must call ITWR to initialize the writing of a specific unit. A successful return from ITWR is then followed by repetitive calls to WRLN to write, in sequence, the lines of data in the unit. A call to TMWR is required after all lines of the unit have been passed to WRLN to write out the last data block of the unit and to release any unused data blocks which were assigned to the over-written unit. WRUN does not add or change any index entry that is associated with the unit written. The index processing modules described in section 2.2.1.3.3 must be called to make any index modification required for the new or changed unit. For random units, if data compression has been specified for the library-section, the data lines will be compressed by WRUN. For independent units, if the independent unit file does not exist, it will be created and allocated by WRUN.

2.2.1.3.2.3.1 ITWR - Initialize to Write

The module ITWR initializes work areas and buffers which will be used subsequently by module Write a Line (WRLN) and Terminate Write (TMWR) to write the account information and data lines of a unit in a PSL library-section file. ITWR reads the first data block of the unit to be written or assigns and initializes a new block for the first data block. It stores the accounting information in the first data block. For a unit typed INDEPENDENT, it allocates the independent unit file. If the independent unit file does not exist, it is created prior to allocation.

a. Program Operations

HIPO diagram 1.3.2.3.1 describes the program operations. In step 1, if the number of the first data block of the unit is not (is equal to zero) in the input index entry, ITWR will call Assign a Block (ASBK) to assign a free block to be used for the first data block. The number of the first data block of the unit is returned to the caller. If the unit is typed INDEPENDENT and the unit did not previously exist (number of first data block equals zero), in step 7, ITWR calls Create a File (CRFL) to create the independent unit file.

b. Data File and Table Descriptions

1. INPUT ARGUMENTS

- 01 FILE-NBR PIC S9(9) COMP.

  - Number assigned to library-section file when it was allocated by ASFL.

- 01 REQUESTED-INDEX-ENTRY.

  05 REQUESTED-UNIT-NAME PIC X(30).

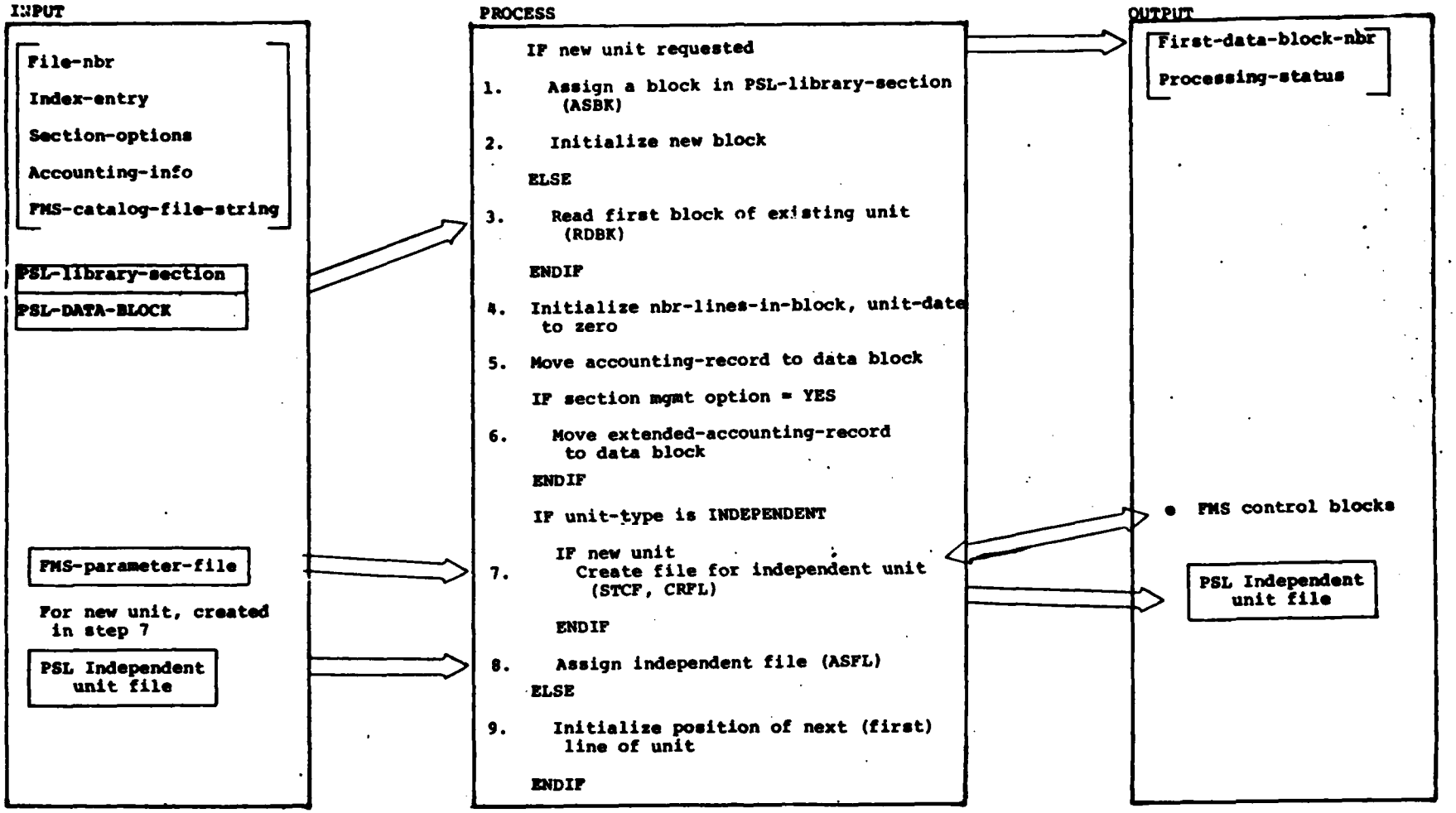
  - Name of unit to be read.

  05 REQUESTED-DATA-BLOCK-PTR PIC S9(9) COMP.

  - Number of first data block of unit to be written. If this field contains zero, a new block will be obtained by ITWR.

Diagram ID: 1.3.2.3.1

Name: ITWR - Initialize to Write



05 REQ-INCLUDING-UNITS-LIST PIC S9(9) COMP.

- Not used.

05 REQUESTED-UNIT-TYPE PIC X(1).

- Unit type code for unit to be read. Code must be one of those defined in the UNIT-TYPE-TABLE.

- 01 SECTION-OPTIONS PIC X(16).

- SECTION-OPTIONS field received from ASFL when library-section to be read was allocated.

- 01 ACCOUNTING-INFO.

- Accounting information to be written in first data block of unit. Fields of accounting information area are described in section .

- 01 FMS-CATALOG-FILE-STRING.

- FMS catalog file string as defined in section for independent unit file when unit typed INDEPENDENT is to be written. Not used for random units.

2. OUTPUT ARGUMENTS

- 01 FIRST-DATA-BLOCK-NBR PIC S9(9) COMP.

- Number of first block of unit to be written. Will contain either the value from the input index entry or the value received from Assign a Block (ASBK) for the new block.

- 01 PROCESSING-STATUS PIC S9(9) COMP.

Return code - zero for normal processing.
- 47 for unable to initialize to read.

3. DATA ITEMS INITIALIZED FOR WRLN AND TMWR

- 01 LINE-COUNTER PIC S9(9) COMP.

- Number of lines written in the current data block. Initial value - zero.

- 01 END-OF-LAST-LINE PIC S9(9) COMP.

  - Character position within the current data block where the last character of the last data record was written. Used only for compressed data. Initial value at start of a first data block - last character position of accounting information or extended accounting information if MGMT option selected for the section. Initial value at start of data blocks other than the first for the unit - zero.

- 01 LINE-POINTER PIC S9(9) COMP.

  - Index to position of last data line record written. Used only for non-compressed, random units. Initial value for first data block - record number which will place the next data line record after the accounting information or extended accounting information if MGMT option selected for the section. Initial value for blocks other than first data block - zero.

c. Branching and Error Conditions

Function Reference	Condition Code	Message Category	Program Action	Note
1	41	PSL	Subsequent processing is bypassed.	1
3	33	PSL	Subsequent processing is bypassed.	2
7	69	ERR	If new block had been assigned, it is released. Subsequent processing is bypassed.	3
7	Any 200	FMS	If new block had been assigned, it is released. Subsequent processing is bypassed.	4
8	6	ADV	If new block had been assigned, it is released. Subsequent processing is bypassed.	5
All	43	PSL	Previous error causes this code to be returned to the caller.	

Notes:

(1) Unable to assign a block. See description of ASBK for detailed list of causes for this condition.

(2) Unable to read a block. See description of RDBK for detailed list of causes for this condition.

(3) Unable to process FMS file options.

(4) Unable to create file. See Sperry Univac 1100 series Executive System Volume 2 Exec Appendix C section C.2 Facility Request Status Codes for error conditions.

(5) Unable to assign a file. See description of ASFL for detailed list of causes for this condition.

2.2.1.3.2.3.2 WRLN - Write a Line

The module WRLN writes a line of data to a unit in a PSL library-section file. Units which may be written are restricted to card-image data in the SOURCE, JOB, LINK, TEST, TEXT, MGMT and PDL sections.

a. Program Operations

HIPO diagram 1.3.2.3.2 describes the program operations. In step 1, the number of leading blanks and the number of characters required for the compressed line are determined by scanning the input line of data to locate the first and last non-blank characters of the line. The number of characters required is:

(Position-of-first-non-blank -
Position-of-last-non-blank + 1) + 4

Two bytes (character positions) are used to store the byte position of the start of the next line and two bytes are used to store the number of leading blanks.

b. Data File and Table Descriptions

1. INPUT ARGUMENTS

- 01 LINE-OF-DATA PIC X(80).

  - The data line which is to be written into the unit being written.

2. OUTPUT ARGUMENTS

- 01 PROCESSING-STATUS PIC S9(9) COMP.

  Return code - zero for line of data written successfully.
  - 44 for error in writing line.

3. OTHER DATA ITEMS

Those data items initialized by ITWR are used by WRLN and updated for use in subsequent calls to WRLN and TMWR. See Section 2.2.1.3.2.3.1 b. for description of these data items.

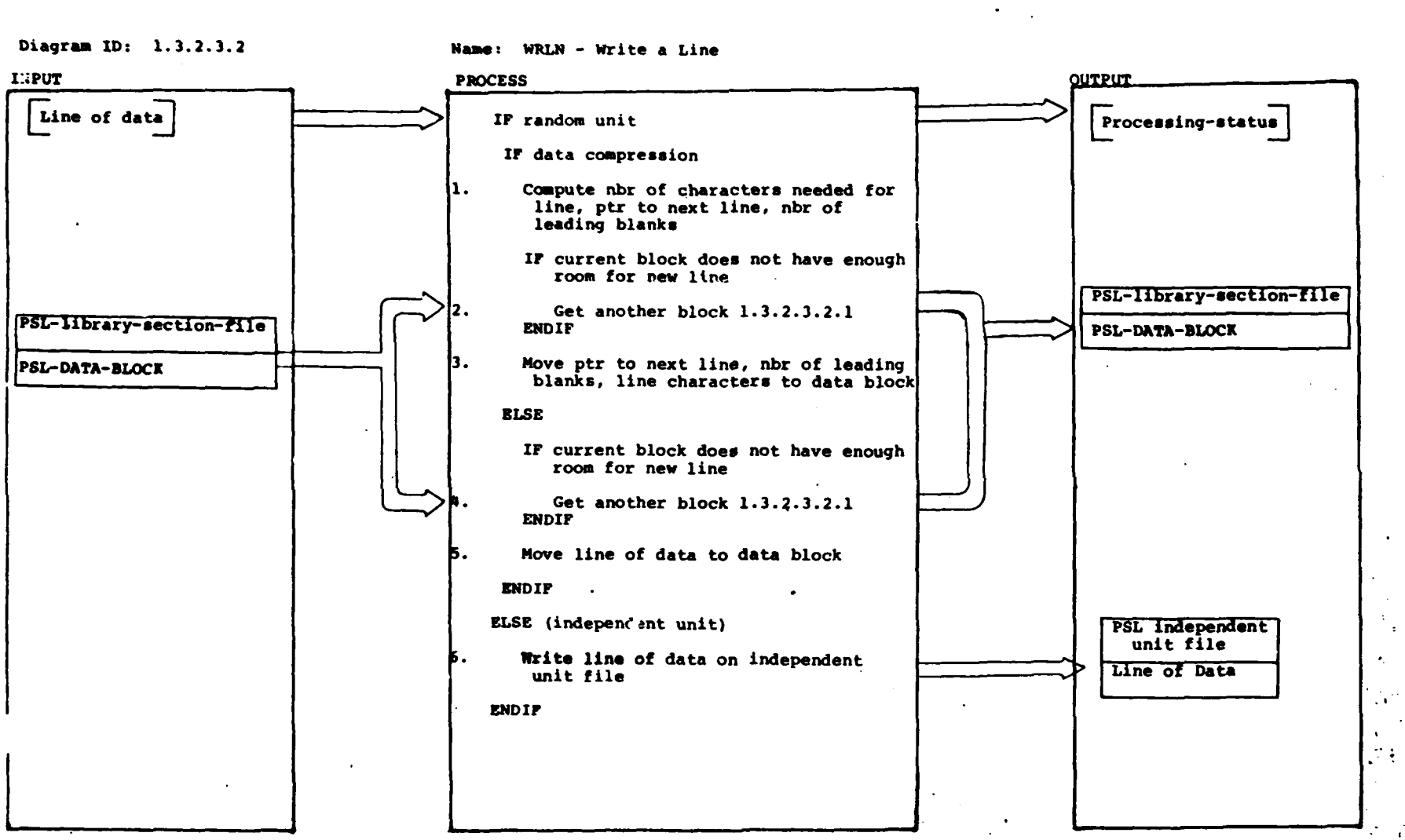
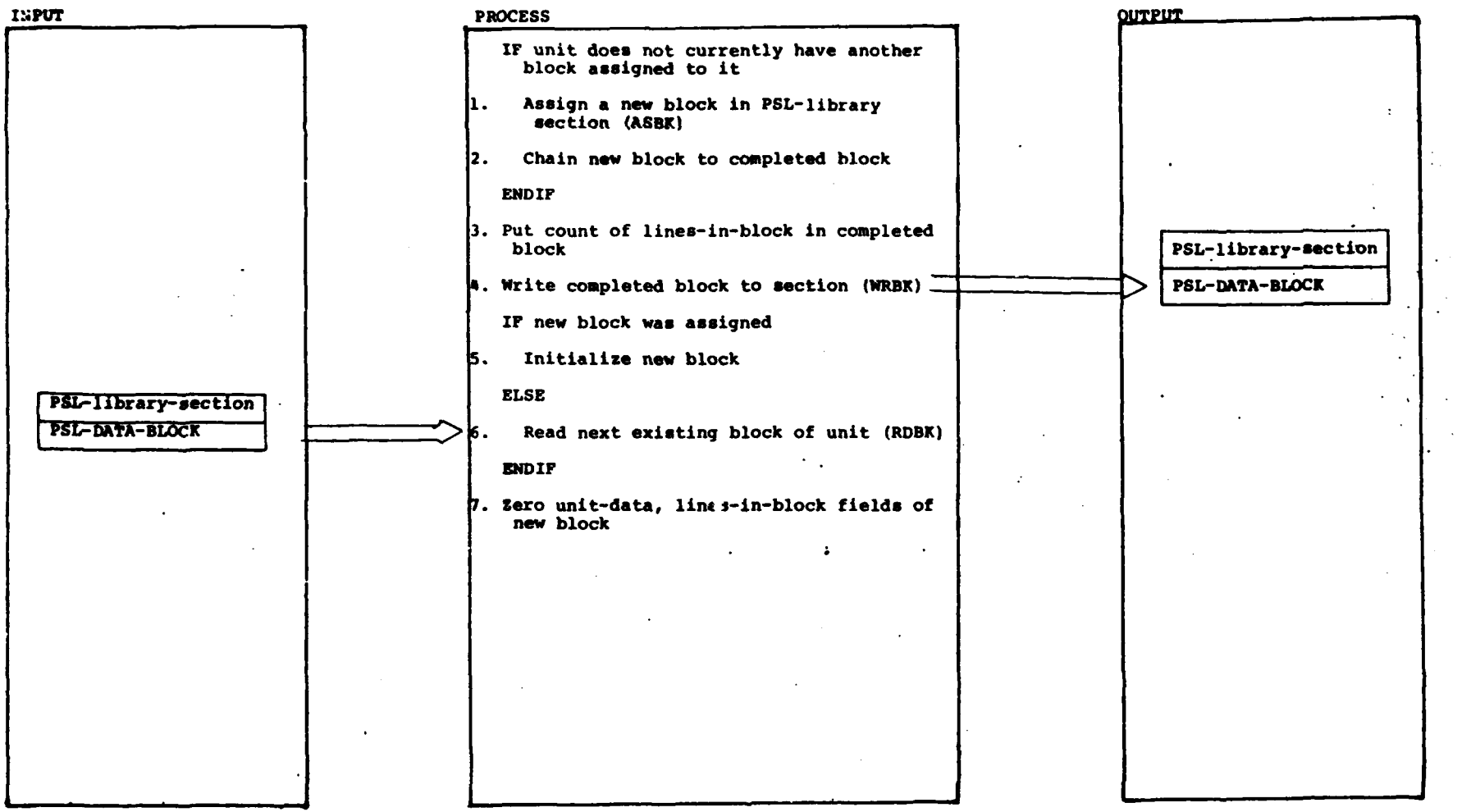


Diagram ID: 1.3.2.3.2
Name: WRLN - Write a Line
INPUT
PROCESS
OUTPUT
Line of data
PSL-library-section-file
PSL-DATA-BLOCK
IF random unit
IF data compression
1. Compute nbr of characters needed for line, ptr to next line, nbr of leading blanks
IF current block does not have enough room for new line
2. Get another block 1.3.2.3.2.1
ENDIF
3. Move ptr to next line, nbr of leading blanks, line characters to data block
ELSE
IF current block does not have enough room for new line
4. Get another block 1.3.2.3.2.1
ENDIF
5. Move line of data to data block
ENDIF
ELSE (independent unit)
6. Write line of data on independent unit file
ENDIF
Processing-status
PSL-library-section-file
PSL-DATA-BLOCK
PSL Independent unit file
Line of Data

Diagram ID: 1.3.2.3.2.1

Name: WRLN - Get another block



c. Branching and Error Conditions

Function Reference	Condition Code	Message Category	Program Action	Note
2(1),4(1)	41	PSL	Line is not written.	1
2(4),4(4)	34	PSL	Line is not written.	2
2(6),4(6)	33	PSL	Line is not written.	3
6	N/A	N/A	Insufficient file space allocated for INDEPENDENT unit file causes program abort.	
All	44	PSL	Error code returned to caller due to previous error.	

Notes:

(1) Unable to assign a block. See description of ASBK for detailed list of causes for this condition.

(2) Unable to write a block. See description of WRBK for detailed list of causes for this condition.

(3) Unable to read a block. See description of RDBK for detailed list of causes for this condition.

2.2.1.3.2.3.3 TMWR - Terminate Write

The module TMWR ends the writing of a unit in a PSL library-section file. This includes writing the last data block to the library-section file, releasing any unused data blocks, and releasing the independent unit file if a unit typed INDEPENDENT was written.

a. Program Operations

HIPO diagram 1.3.2.3.3 describes the program operations. No index processing is performed by TMWR. If the newly written unit is to be used by subsequent PSL function processors, the index processing modules must be called to update the index entry to reflect the new position of the data blocks for the unit in the library-section file.

b. Data File and Table Descriptions

All significant data items are described in Section 2.2.1.3.3.1 with the module ITWR.

c. Branching and Error Conditions

Function Reference	Condition Code	Message Category	Program Action	Note
1	42	PSL	NONE	1
4	34	PSL	NONE	2
5	37	PSL	NONE	3

Notes:

(1) Unable to release excess blocks. See description of RLBK for details of causes for this condition.

(2) Unable to write a block. See description of WRBK for details of causes for this condition.

(3) Unable to release a file. See description of RLFL for details of causes for this condition.

Diagram ID: 1.3.2.3.3

Name: TMWR - Terminate Write

INPUT

PROCESS

IF there are any unused blocks left for the unit being written

1. Release excess blocks (RLBK)

ENDIF

2. Move last-block-flag (0) to next-block-ptr of last block

3. Put nbr-of-lines-in block in last block

4. Write last block of unit (WRBK)

IF independent unit

5. Release independent unit file (RLFL)

ENDIF

OUTPUT

PSL-library-section

Free block switches

Date blocks

2.2.1.3.3 Index I/O

Index I/O is performed to add, find, read, change and delete an index entry in a PSL section file. The program modules which perform these operations are described below.

2.2.1.3.3.1 ADXE - Add an Index Entry

The module ADXE adds a PSL INDEX ENTRY, alphabetically, to an INDEX BLOCK of the section file. The ADXE processing is responsible for finding the appropriate INDEX BLOCK to which to add the new INDEX ENTRY. The new INDEX ENTRY may be added to an existing INDEX BLOCK or to a new INDEX BLOCK. The new INDEX BLOCK results from either having no space to add another entry or the need to insert an entry within the alphabetical range of a full INDEX BLOCK. It is assumed that prior to ADXE processing, the module ASFL has been called to allocate the PSL section file.

Modules that call the module ADXE are responsible for setting the USE-DIRECTORY-SW which indicates the starting block for the ADXE processing, either the CONTROL BLOCK or the first INDEX BLOCK.

a. Program Operations

HIPO diagram 1.3.3.1 describes the operation performed. In step 2, if the directory block of a PSL section file is to be read by the ADXE processing, the PSL CONTROL BLOCK must be read to obtain the location of the directory block. In step 7, the ADXE processing compares unit names of each index entry to find the proper position for the new entry. In step 8, when the new unit name is alphabetically within range of the existing unit name in a particular index block and the existing entry points to a lower index block, then ADXE processing must inspect the lower index block for proper position. In step 10, if the new index entry is to be placed in an index block which is already full then the full block has to be split and rewritten to accommodate the new entry. Index blocks that are not full may have the new entry added or inserted before the block is rewritten.

b. Data File and Table Description

1. INPUT ARGUMENTS

- FILE-NBR PIC S9(9) COMP.
  - number of file code, within the allocation table to which file has been assigned.
- REQUESTED-BLOCK-NBR PIC S9(9) COMP.
  - index block where processing is to begin.

Diagram ID: 1.3.3.1

Name: ADXE - Add an Index Entry

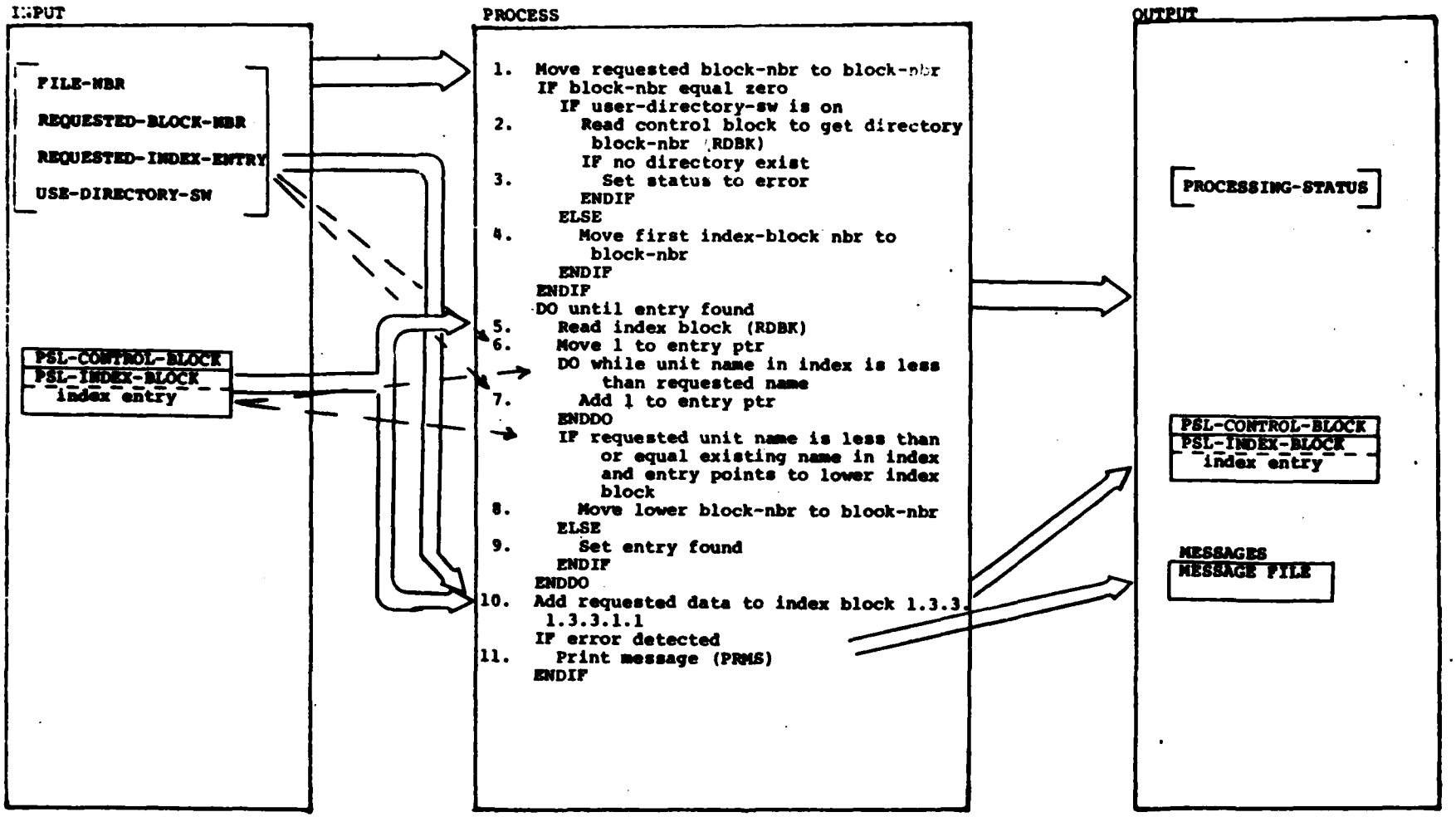
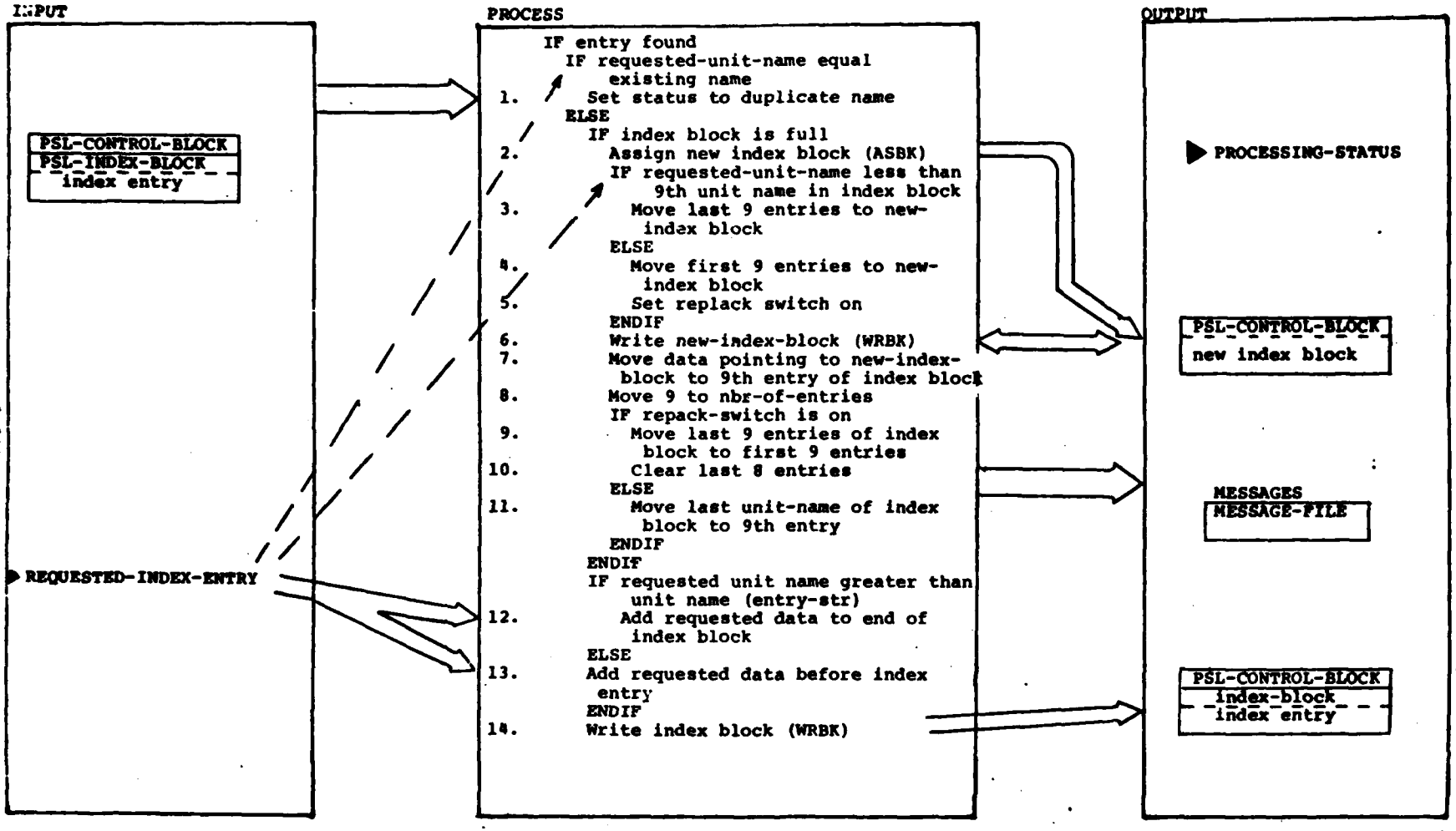


Diagram ID: 1.3.3.1.1

Name: ADXE - Add Requested Data to Index Block



- REQUESTED-INDEX-ENTRY

  - see section for description of the data items of the new index entry

- USE-DIRECTORY-SW PIC S9(9) COMP.

  - indicates that the directory block location must be obtained from the CONTROL BLOCK

2. OUTPUT ARGUMENT

- PROCESSING-STATUS PIC S9(9) COMP.

  Return-code - zero for normal status
  - error code indicating cause of failure

c. Branching and Error Conditions

Function Reference	Condition Code	Message Category	Program Action	Note
2,5	33	PSL	Subsequent processing is bypassed	
3	27	PSL	Subsequent processing is bypassed	
10	25	ERR	Subsequent processing is bypassed	
10	41,34	PSL	Subsequent processing is bypassed	1,2
11	24	ERR	Subsequent processing is bypassed	

Notes:

(1) See description of module ASBK for conditions which cause "Unable to assign block" condition.

(2) See description of module WRBK for conditions which cause "Unable to write block" condition.

2.2.1.3.3.2 CHXE - Change an Index

The module CHXE changes the data pointers and the UNIT TYPE of an INDEX ENTRY in a PSL INDEX BLOCK. It is assumed that the module ASFL has been called to allocate the section file. Modules that call the CHXE module are responsible for setting the USE-DIRECTORY-SW which indicates the starting block for CHXE processing, either the CONTROL BLOCK or the first INDEX BLOCK.

a. Program Operations

HIPO diagram 1.3.3.2 describes the operation performed. Most of the CHXE processing is similar to module FDXE 2.2.1.3.3.3. In step 8, when the matching unit name is located, the data items of the INDEX ENTRY are replaced by the REQUESTED-INDEX-ENTRY. In step 9, if an error occurs and the INDEX BLOCK is not written, the INDEX BLOCK remains unchanged.

b. Data File and Table Descriptions

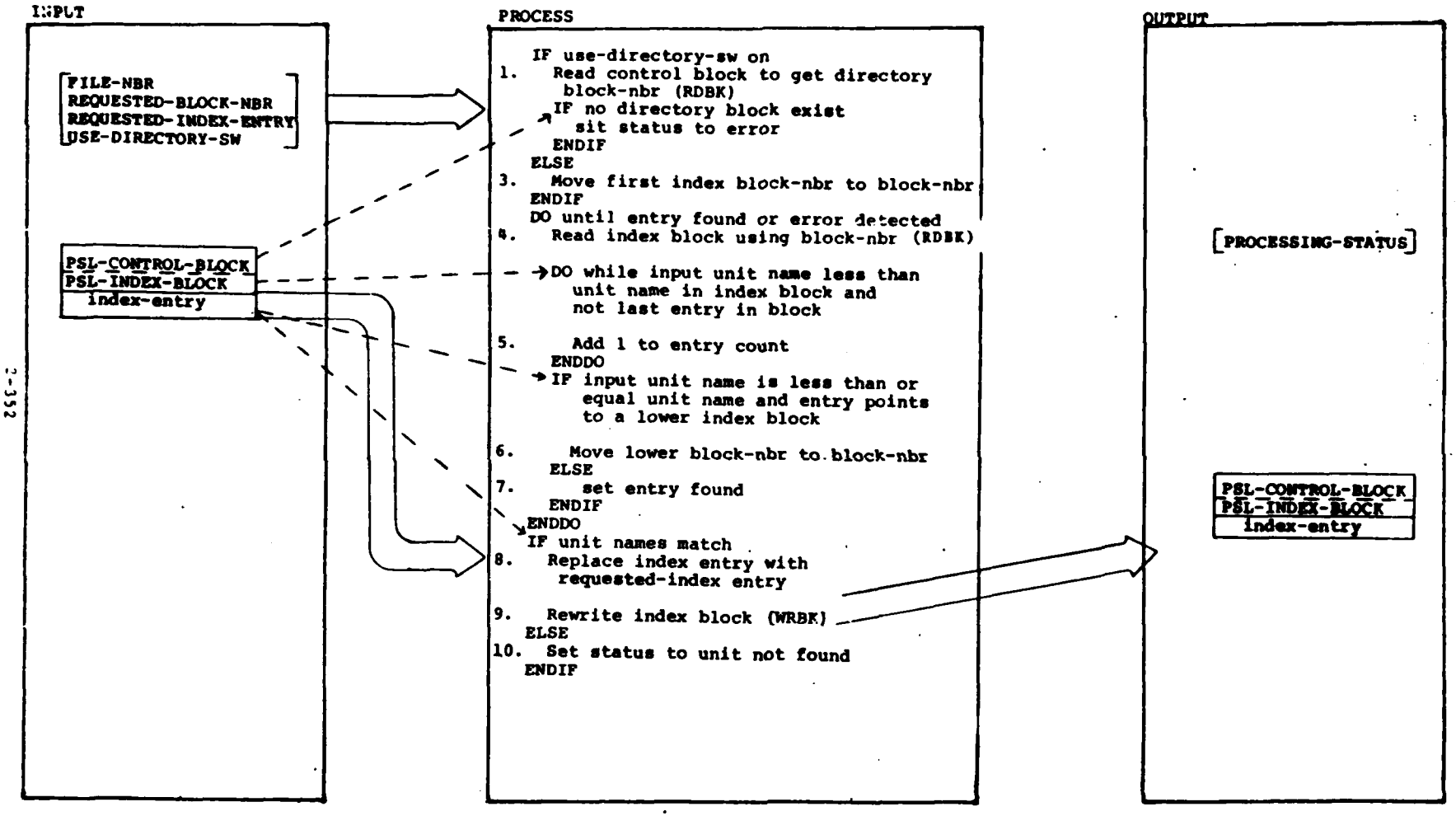
There are no significant files or tables in the module.

1. INPUT ARGUMENTS

- FILE-NBR PIC S9(9) COMP.
  - file number of section file where INDEX is to be searched
- REQUESTED-BLOCK-NBR PIC S9(9) COMP.
  - starting INDEX BLOCK of CHXE processing
- REQUESTED-INDEX-ENTRY
  - see section for definition
  - data items to replace data in existing INDEX ENTRY
- USE-DIRECTORY-SW PIC S9(9) COMP.
  - indicates that the DIRECTORY BLOCK location must be obtained from the CONTROL BLOCK

Diagram ID: 1.3.3.2

Name: CHXE - Change an Index Entry



2. OUTPUT ARGUMENTS

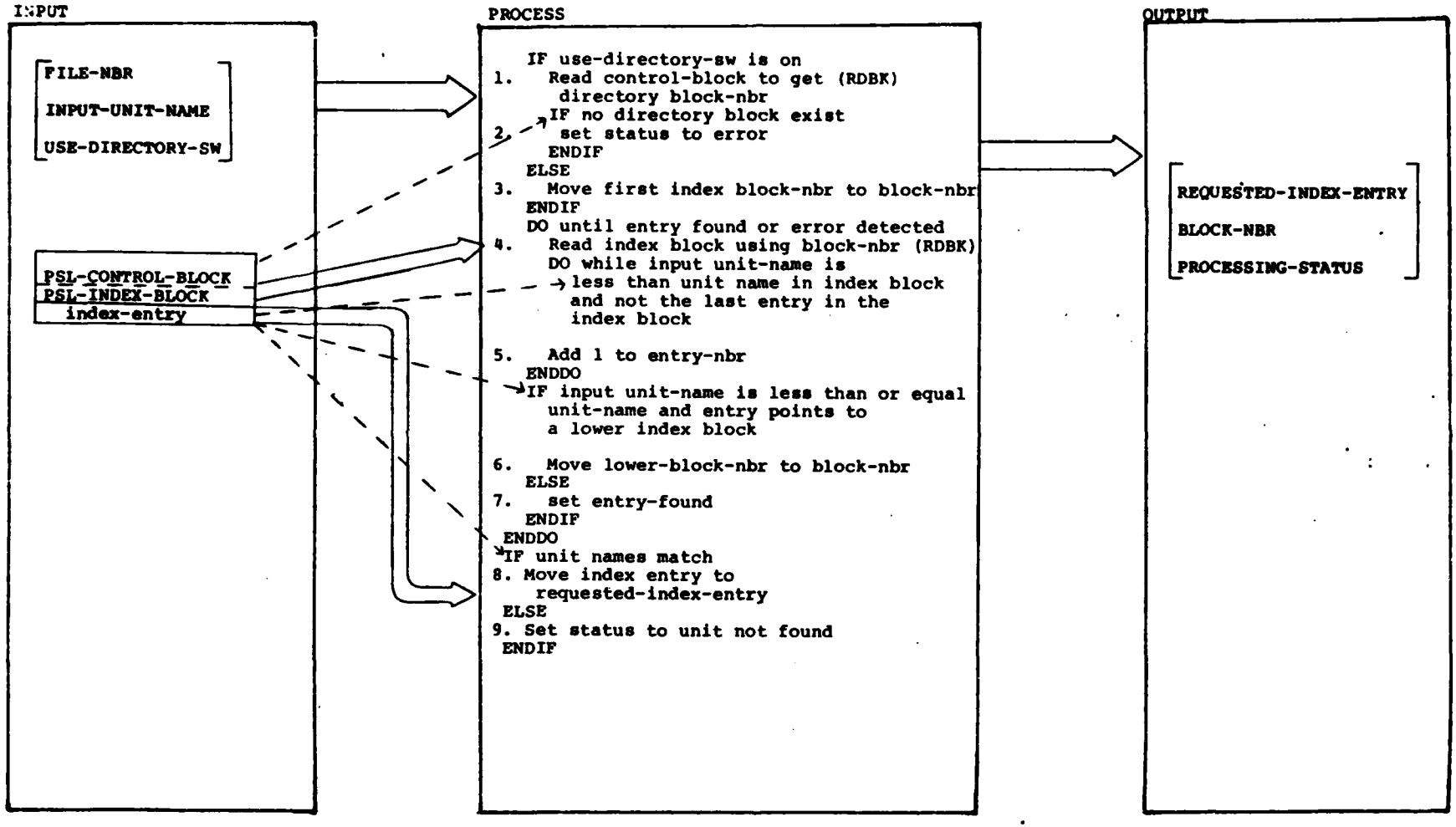
- PROCESSING-STATUS PIC S9(9) COMP.
  (return code)
  - zero for normal status
  - error code indicating cause of failure

c. Branching and Error Conditions

Function Reference	Condition Code	Message Category	Program Action	Note
1,4	33	PSL	Subsequent processing bypassed	
1,4	29,7	ERR	Subsequent processing bypassed	
2	27	PSL	Subsequent processing bypassed	
9	34	PSL	Subsequent processing bypassed	
10	26	ERR	Subsequent processing bypassed	

Diagram ID: 1.3.3.3

Name: FDXE - Find an Index Entry



2.2.1.3.3.3 FDXE - Find an Index Entry

The module FDXE accesses the PSL CONTROL BLOCK and the PSL INDEX BLOCK of a PSL Section File to find data unit. The module FDXE is called by various PSL modules. In normal processing the module ASFL is called prior to calling the FDXE module. The module ASFL allocates the PSL Section File for access by the FDXE module.

Modules that call the FDXE module are responsible for setting the USE-DIRECTORY-SW which indicates the starting block of the FDXE processing, either at the PSL CONTROL BLOCK or the first PSL INDEX BLOCK.

a. Program Operations

HIPO diagram 1.3.3.3 describes the operation performed. In step 1, if a directory block of a PSL Section File is to be read by the FDXE processing, the PSL CONTROL BLOCK must be read to obtain the location of the directory block. In step 5, the PSL INDEX BLOCK may contain as many as 17 entries. The number of actual entries is stored at the beginning of the index block. In step 6, an index entry may contain the location of another PSL INDEX BLOCK (refer to module ADXE).

The FDXE processing reads through the index block looking for matching unit names or a unit name which has a higher collating value than the request unit name. If index blocks are chained to other index blocks the FDXE processing will read through the chained blocks in order to satisfy the previous condition.

b. Data File and Table Descriptions

There are no significant files or tables in this module.

1. INPUT ARGUMENT

- FILE-NBR PIC S9(9) COMP.
  - file number of section file where INDEX BLOCK is to be searched
- INPUT-UNIT-NAME PIC X(30).
  - name of unit to be found in INDEX BLOCK

- USE-DIRECTORY-SW PIC S9(9) COMP.

  - indicates that the DIRECTORY BLOCK location must be obtained from the Control Block stored data

2. OUTPUT ARGUMENTS

- REQUESTED-INDEX-ENTRY

  - see section for definition data items that compose an INDEX ENTRY for a data unit in a PSL INDEX BLOCK

- BLOCK-NBR PIC S9(9) COMP.

  - INDEX BLOCK where FDXE processing stopped

- PROCESSING-STATUS PIC S9(9) COMP.

  Return code - zero for normal status
  - error code indicating cause of failure

c. Branching and Error Conditions

Function Refernece	Condition Code	Message Category	Program Action	Note
1,4	33,31	PSL	Subsequent processing is bypassed	
2	27,31	PSL	Subsequent processing is bypassed	
8	0	PSL	NORMAL-PROCESSING	
9	26	PSL	Subsequent processing continues	

2.2.1.3.3.4 DLXE - Delete an Index Entry

The module DLXE deletes an INDEX ENTRY from the INDEX BLOCK of a PSL section file. It is assumed that the module ASFL has been called to allocate the section file. Modules that call the DLXE module are responsible for setting the USE-DIRECTORY-SW which indicates the starting block for DLXE processing, either the CONTROL BLOCK or the first INDEX BLOCK.

a. Program Operations

HIPO diagram 1.3.3.4 describes the oeprations performed. Most of the DLXE processing is similar to module FDXE 2.2.1.3.3.3. In step 8, when the matching unit name is located, the INDEX ENTRY is cleared, the number of entries in the block is reduced, and the block is compressed and re-written.

b. Data File and Table Descriptions

There are no significant files or tables in this module.

1. INPUT ARGUMENTS

- FILE-NBR PIC S9(9) COMP.

  - file number of section file where INDEX BLOCK is to be searched

- BLOCK-NBR PIC S9(9) COMP.

  - INDEX BLOCK initialized to zero

- INPUT-UNIT-NAME PIC X(30).

  - name of unit in INDEX BLOCK to be deleted

- USE-DIRECTORY-SW PIC S9(9) COMP.

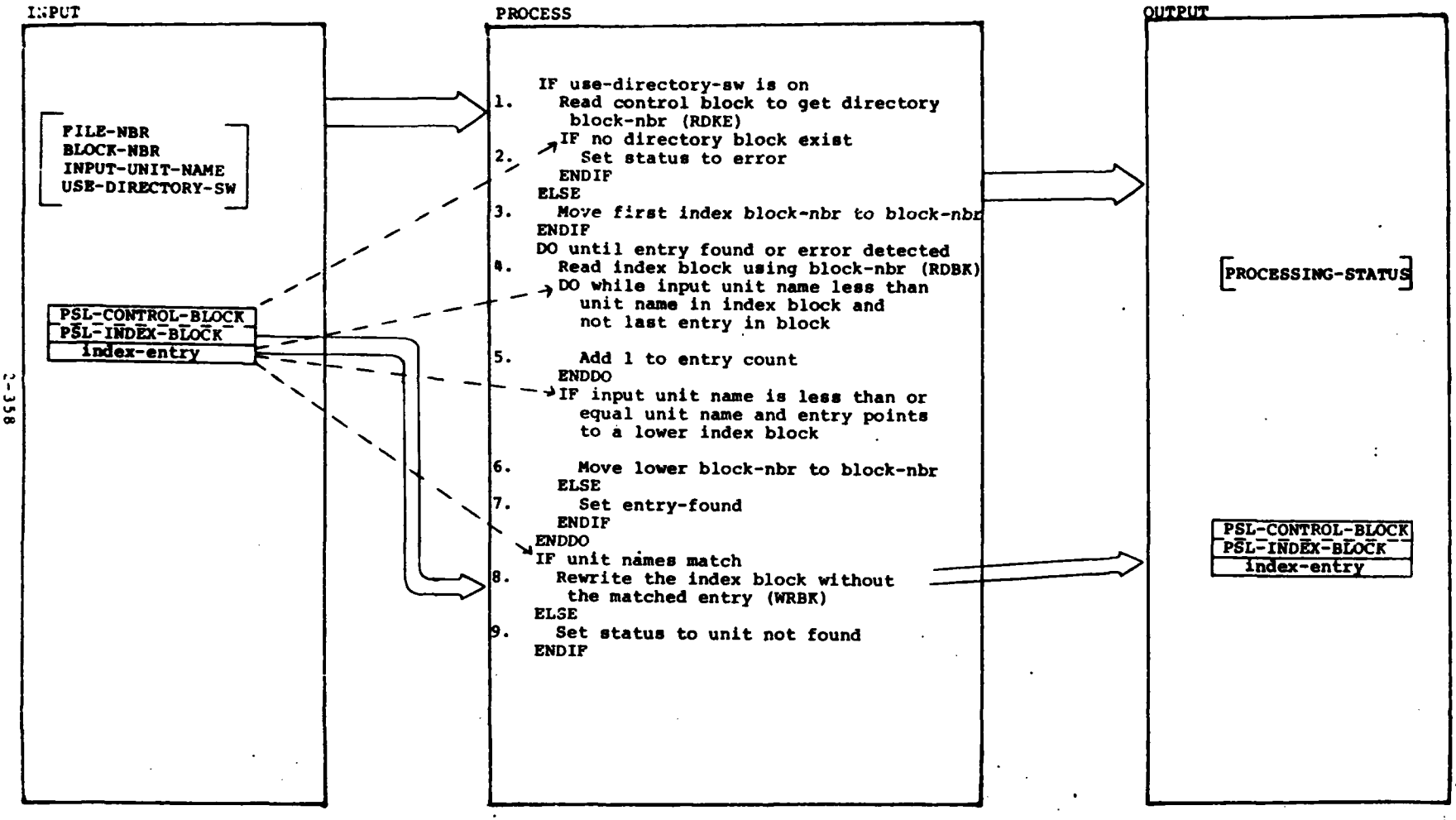
2. OUTPUT ARGUMENT

- PROCESSING-STATUS PIC S9(9) COMP.

  Return code - zero for normal status
  - error code indicating cause of failure

Diagram ID: 1.3.3.4

Name: DLXE - Delete an Index Entry



c. Branching and Error Conditions

Function Reference	Condition Code	Message Category	Program Action	Note
1,4	33	PSL	Subsequent processing bypassed	
1,4	7	ERR	Processing terminates	
2	27	PSL	Subsequent processing bypassed	
2	7	ERR	Processing terminates	
7	26	ERR	Subsequent processing bypassed	
8	26	ERR	Subsequent processing bypassed	
8	34	PSL	INDEX BLOCK remains unchanged. Subsequent processing bypassed	1

Note:

(1) See description of module WRBK for conditions which cause "unable to write block" conditions.

2.2.1.3.3.5 RDXX - Read an Index Entry

The module RDXX obtains a PSL INDEX ENTRY by sequentially reading through an INDEX BLOCK. Each call to the module RDXX will obtain the next INDEX ENTRY of the INDEX BLOCK in the PSL section file. The RDXX processing preserves the pointers to the section file and INDEX BLOCKS for subsequent calls. It is assumed that the module ASFL has been called to allocate the section file. Modules that call the module RDXX are responsible for setting the USE-DIRECTORY-SW, which indicates the starting block for RDXX processing, either the CONTROL BLOCK or the first INDEX BLOCK.

a. Program Operations

HIPO diagram 1.3.3.5 describes the operation performed. In step 1, if a directory block of a PSL section file is to be read by the RDXX processing, the PSL CONTROL BLOCK must be read to obtain the location of the directory block. In step 3, if the INDEX BLOCK contains no entries or the last entry has been obtained, then a stored block pointer is retrieved from the STACK-ARRAY and the BUFFER ARRAY is searched for the retrieved block pointer. If the INDEX BLOCK is in the buffer array, it is accessed, otherwise, the INDEX BLOCK is read via the module RDBK. In step 11, if an INDEX ENTRY points to a lower INDEX BLOCK, the block location and entry pointer are stored in the stack-array and the index block itself is stored in the buffer-array. Then, the lower INDEX BLOCK is read.

b. Data File and Table Descriptions

- 01 BUFFER-ARRAY

    05 BUFFERS OCCURS 4 TIMES

        - INDEXED BY SEARCH-PTR

        10 BUFFER.

            15 BUFFER-BLOCK-NBR PIC S9(9) COMP.

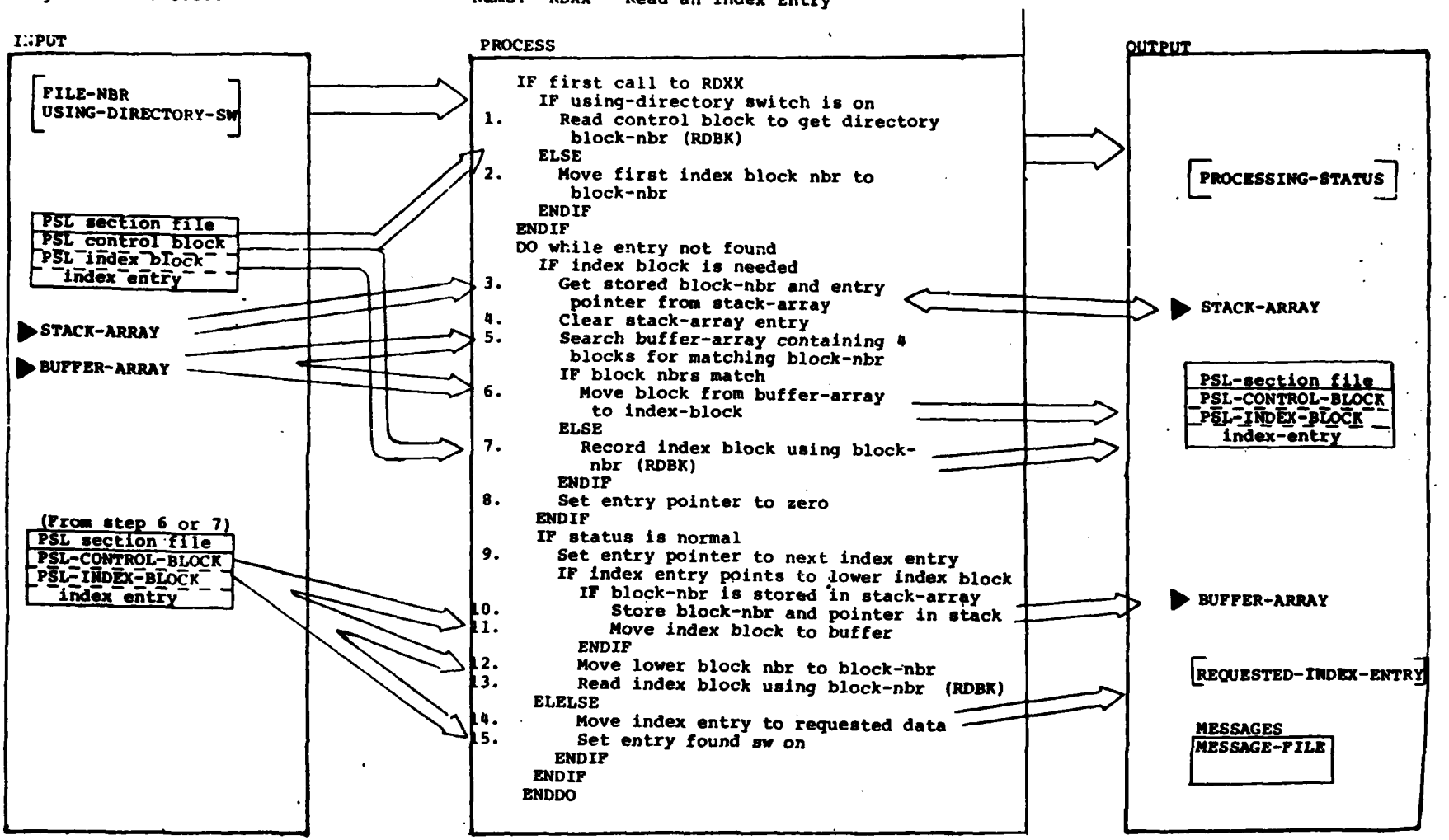
                - location of INDEX-BLOCK

        10 FILLER PIC X(762).

            - stored INDEX-BLOCK

Diagram ID: 1.3.3.5

Name: RDXX - Read an Index Entry



- 01 STACK-ARRAY

  05 STACK-ENTRY OCCURS 100 TIMES

    10 BLOCK-NBR-IN-STACK-ENTRY PIC S9(9) COMP.

    - location of INDEX BLOCK

    10 INDEX-PTR-IN-STACK-ENTRY PIC S9(9) COMP.

    - pointer to INDEX ENTRY

1. INPUT ARGUMENTS

   - FILE-NBR PIC S9(9) COMP.

     - file number of the section file where INDEX BLOCK is to be searched

   - USING-DIRECTORY-SW PIC S9(9) COMP.

     - indicates that the directory block location must be obtained from the CONTROL BLOCK stored data

2. OUTPUT ARGUMENTS

   - REQUESTED-INDEX-ENTRY

     - see section 3 for definition
     - data items correspond to INDEX ENTRY data item

   - PROCESSING-STATUS PIC S9(9) COMP.

     Return code - zero for normal block
                 - error code indicating cause of failure

c. Branching and Error Conditions

Function Reference	Condition Code	Message Category	Program Action	Note
1	27,33	PSL	Subsequent processing bypassed	
3	33	PSL	Subsequent processing bypassed	
3	50	ERR	Subsequent processing bypassed	
7	33	PSL	Subsequent processing bypassed	
13	33	PSL	Subsequent processing bypassed	

Note:

(1) The above error conditions will cause the module RDXX a return code value 51 "Unable to read INDEX".

2.2.1.4 Executive Operating System Interface Routines

The modules in this subdivision of the PSL system are used to interface with the Univac 1100 Series Operating System for communicating with the file control system.

2.2.1.4.1 Setup Parameter Routines

Except for the Function "obtain user ID" (OBUSE), all functions resulting in the execution of an executive request to the operating system is setup by the multi-entry point COBOL module ALLOCATOR. This module is described as follows:

2.2.1.4.1.1 ALLOCATOR - Process Parameters

The functions at the various entry points in the module ALLOCATOR build the appropriate input needed by the assembly language modules to invoke the required executive request.

a. Program Operations

HIPO diagram 1.4.1.1 through 1.4.1.7 describe the program operation for each entry point routine within the module ALLOCATOR.

b. Data File and Table Descriptions

```
• 01  FMS-PARAMETER-RECORD.

      05  PRIME-KEYWORD.
          10  PRIME-KW-CHARACTER OCCURS 12 TIMES
                                        PIC X .
      05  FILLERD REDEFINES PRIME-KEYWORD.
          10  SHORT-PRIME-KEYWORD      PIC X(4) .
          10  FILLER                   PIC X(8) .
      05  SUB-KEYWORD.
          10  SUB-KW-CHARACTER OCCURS 12 TIMES
                                        PIC X .
      05  FILLERE REDEFINES SUB-KEYWORD.
          10  SHORT-SUB-KEYWORD        PIC X(4) .
          10  FILLER                   PIC X(8) .
      05  LENGTH-OF-VALUE              PIC S9(9) COMP.
      05  KEYWORD-VALUE.
          10  KW-VALUE-CHARACTER OCCURS 48 TIMES
                                        PIC X .
      05  FILLERF REDEFINES KEYWORD-VALUE.
          10  SHORT-KEYWORD-VALUE      PIC X(4) .
          10  FILLER                   PIC X(44) .
      05  LAST-PRIME-KW-SW             PIC S9(9) COMP.
          88  LAST-PRIME-SW                VALUE 1.
      05  LAST-SUB-KW-SW               PIC S9(9) COMP.
          88  LAST-SUB-KW                  VALUE 1.
      05  LAST-KW-VALUE-SW             PIC S9(9) COMP.
          88  LAST-KW-VALUE                VALUE 1.

• 01  ERROR-MESSAGE-TABLE.
      05  FILLER                       PIC X(54)
          VALUE 'REQUEST REJECTED-UNDEFINED ERROR'.
      05  FILLER                       PIC X(54)
          VALUE 'REQUEST REJECTED-UNDEFINED ERROR'.
      05  FILLER                       PIC X(54)
          VALUE 'REQUEST REJECTED-UNDEFINED ERROR'.
```

```
05  FILLER                          PIC X(54)
    VALUE 'REQUEST REJECTED-UNDEFINED ERROR'.
05  FILLER                          PIC X(54)
    VALUE 'REQUEST REJECTED'.
05  FILLER                          PIC X(54)
    VALUE 'REQUEST REJECTED-F CYCLE CONFLICT'.
05  FILLER                          PIC X(54)
    VALUE 'REQUEST REJECTED-FILE DISABLED'.
05  FILLER                          PIC X(54)
    VALUE 'REQUEST REJECTED-FILE DISABLED'.
05  FILLER                          PIC X(54)
    VALUE 'REQUEST REJECTED-FILE DISABLED'.
05  FILLER                          PIC X(54)
    VALUE 'REQUEST REJECTED-EQUIPMENT REQUESTED IS DOWN'.
05  FILLER                          PIC X(54)
    VALUE 'REQUEST REJECTED-WRITE ONLY FILE'.
05  FILLER                          PIC X(54)
    VALUE 'REQUEST REJECTED-READ ONLY FILE'.
05  FILLER                          PIC X(54)
    VALUE 'REQUEST REJECTED-EQUIPMENT IS TAPE'.
05  FILLER                          PIC X(54)
    VALUE 'REQUEST REJECTED-PROJECT ID INCORRECT'.
05  FILLER                          PIC X(54)
  VALUE 'REQUEST REJECTED-UNASSIGNED FILE NOT DECATALOGUED'.
05  FILLER                          PIC X(54)
    VALUE 'REQUEST REJECTED-FILE BUSY,TRY AGAIN'.
05  FILLER                          PIC X(54)
    VALUE 'REQUEST REJECTED-FILE BUSY,TRY AGAIN'.
05  FILLER                          PIC X(54)
    VALUE 'REQUEST REJECTED-OPTION CONFLICT'.
05  FILLER                          PIC X(54)
    VALUE 'REQUEST REJECTED-FACILITY BUSY,TRY AGAIN'.
05  FILLER                          PIC X(54)
    VALUE 'REQUEST REJECTED-FILE ROLLED OUT'.
05  FILLER                          PIC X(54)
    VALUE 'REQUEST REJECTED-INCORRECT REEL/PACK-ID'.
05  FILLER                          PIC X(54)
  VALUE 'REQUEST REJECTED-FILENAME NOT IN MASTER DIRECTORY'.
05  FILLER                          PIC X(54)
    VALUE 'REQUEST REJECTED-NO WRITE KEYS EXIST'.
05  FILLER                          PIC X(54)
    VALUE 'REQUEST REJECTED-NO READ KEYS EXIST'.
05  FILLER                          PIC X(54)
    VALUE 'REQUEST REJECTED-INCORRECT KEY'.
05  FILLER                          PIC X(54)
    VALUE 'REQUEST REJECTED-INCORRECT KEY'.
05  FILLER                          PIC X(54)
    VALUE 'REQUEST REJECTED-INCORRECT KEY'.
05  FILLER                          PIC X(54)
    VALUE 'REQUEST REJECTED-INCORRECT KEY'.
```

```
05  FILLER                          PIC X(54)
    VALUE 'REQUEST REJECTED-FILE ALREADY EXCLUSIVE'.
05  FILLER                          PIC X(54)
    VALUE 'REQUEST REJECTED-INTERNAL FILENAME NOT UNIQUE'.
05  FILLER                          PIC X(54)
    VALUE 'REQUEST REJECTED-UNDEFINED ERROR'.
05  FILLER                          PIC X(54)
    VALUE 'REQUEST REJECTED-EQUIPMENT NOT COMPATIBLE'.
05  FILLER                          PIC X(54)
    VALUE 'REQUEST REJECTED-FILE PREVOUSLY CATALOGUED'.
05  FILLER                          PIC X(54)
    VALUE 'REQUEST REJECTED-FILE ALREADY ASSIGNED'.
05  FILLER                          PIC X(54)
    VALUE 'REQUEST REJECTED-HARDWARE NOT PART OF SYSTEM'.
05  FILLER                          PIC X(54)
    VALUE 'REQUEST NOT ACCEPTED'.
```

- 01 ERMSG-TABLE-ENTRY PIC X(54) OCCURS 36 TIMES.

1. INPUT ARGUMENTS

FMS-PARAMETER-RECORD

- See data file for description
- Contains track sizes for files

- 01 FMS-CATALOG-FILE-STRING

```
05  FMS-CATALOG-FILE-ENTRIES.
    10  FMS-PROJECT-ENTRY.
        15  FMS-PROJECT-NAME        PIC X(12) .
        15  FMS-PROJECT-PASSWORD    PIC X(12) .

    10  FMS-LIB-SEC-ENTRY.
        15  FMS-LIB-SEC-NAME        PIC X(12) .
        15  FMS-LIB-SEC-PASSWORD    PIC X(12) .

    10  FMS-UNIT-ENTRY.
        15  FMS-UNIT-NAME           PIC X(12) .
        15  FMS-UNIT-PASSWORD       PIC X(12) .
05  FMS-END-OF-CATALOG-FILE-STRING
                                    PIC X(12)
                                    VALUE HIGH-VALUES.
```

- Contains the external filename

FILE CODE PIC X(2)

- Contains the internal filename

2. OUTPUT ARGUMENTS

```
PROCESSING-STATUS          PIC S9(9) COMP.
```

- 01 TRACKS-ASSIGNED REDEFINES PROCESSING-STATUS.
  - 05 INITIAL-TRACKS PIC 9(4) COMP.
  - 05 MAX-TRACKS PIC 9(4) COMP.

Return code - zero for normal status **QYFLE returns zero for tracks assigned for abnormal status
- error code indicating cause of request rejection **QYFLE returns track sizes assigned for normal status

- 01 EXECUTIVE-REQUEST-STATEMENT.
  - 05 EXEC-OPERATOR PIC X(5) VALUE SPACES.
  - 05 EXEC-OPTION PIC X(2) VALUE SPACES.
  - 05 FILLER PIC X(1) VALUE SPACE.
  - 05 EXEC-FILE-NAME.
    - 10 EXEC-FILE-CHAR PIC X(1) OCCURS 26 TIMES.
  - 05 EXEC-OPERANDS.
    - 10 EXEC-PARMS.
      - 15 EXEC-TYPE PIC X(2).
      - 15 EXEC-RESRV-LGTH.
        - 20 EXEC-RESRV-LENGTH PIC X(1) OCCURS 6 TIMES.
      - 15 EXEC-MAX-LENGTH.
        - 20 EXEC-MAXLGT2 PIC X(1) OCCURS 4 TIMES.
  - 05 FILLER PIC X(5) VALUE ' . . '.

- Used as input to called executive request routine

c. Branching and Error Conditions

Function Reference	Condition Code	Message Category	Program Action	Note
	200-236	FMS	Subsequent processing continues	1

Note

(1) See Sperry Univac 1100 Series Executive System Diagnostic Messages and Status Codes 'Facility Status Bits Table' for specific conditions which cause the error.

Diagram ID: 1.4.1.1 Name: ALFLE - Allocate a File Description: Routine within module ALLOCATOR

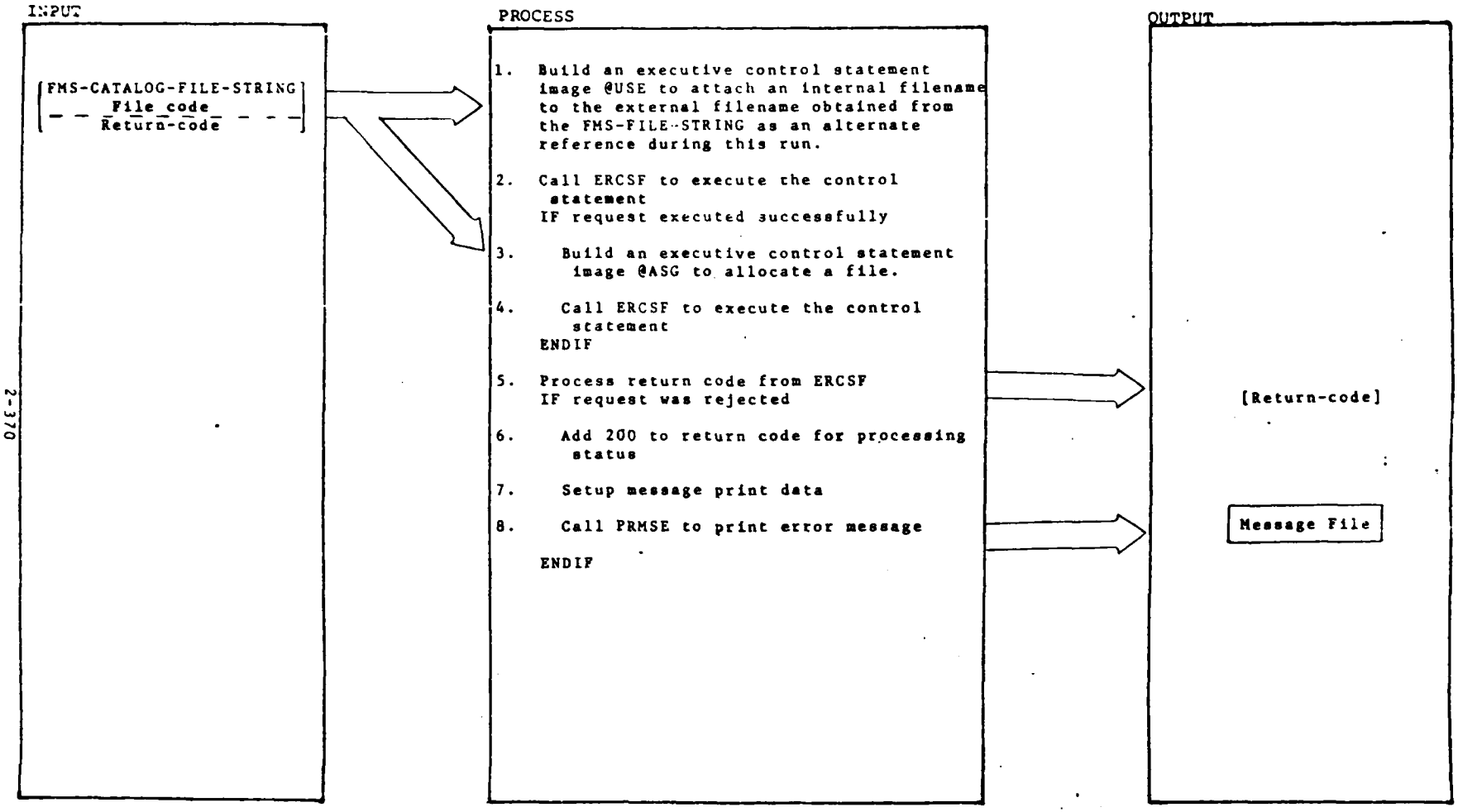
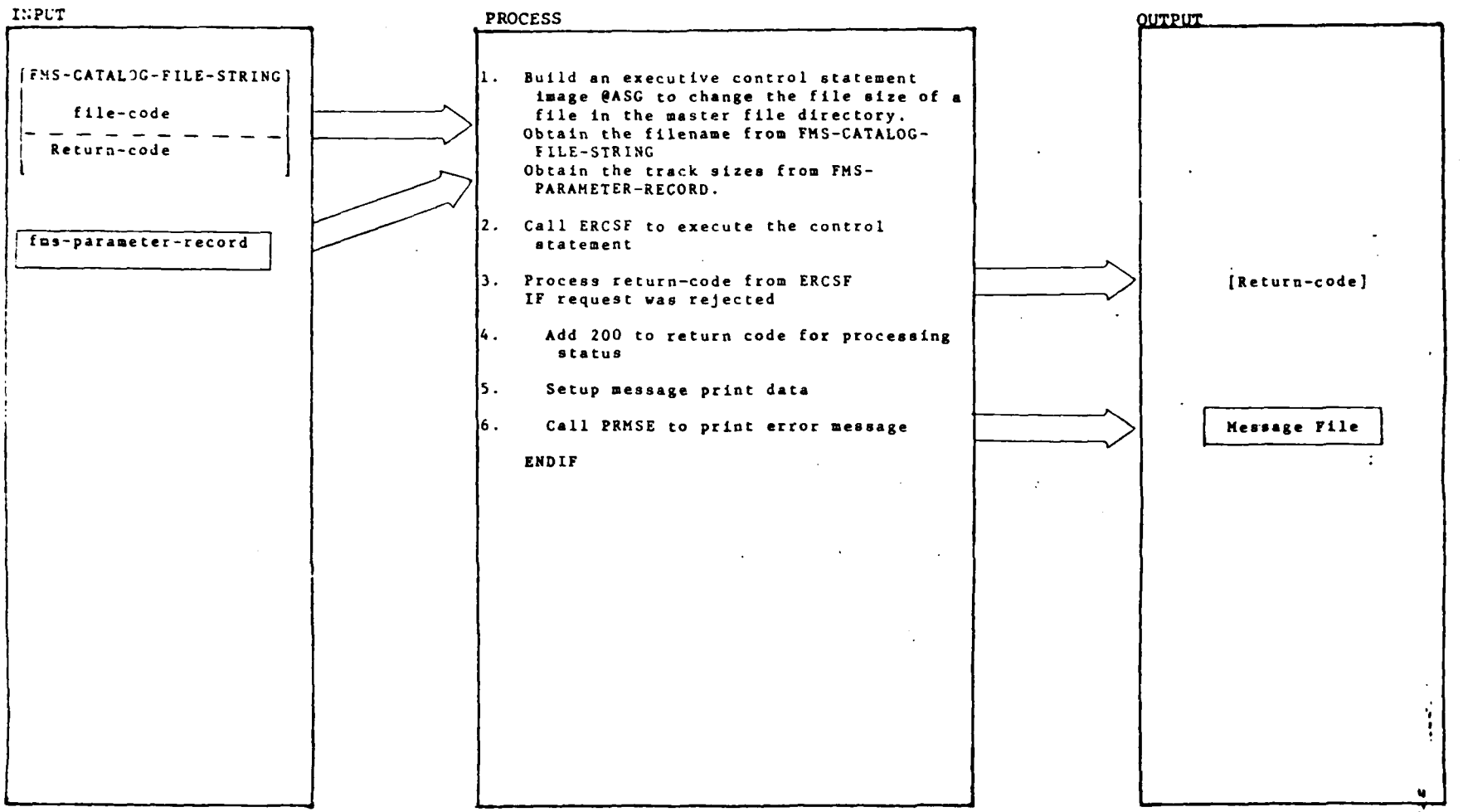


Diagram ID: 1.4.1.2

Name: CHFLE - Change a File

Description: Routine within module ALLOCATOR



IBM-G-4

Diagram ID: 1.4.1.3

Name: CRFLE - Create a Catalog

Description: Routine within module ALLOCATOR

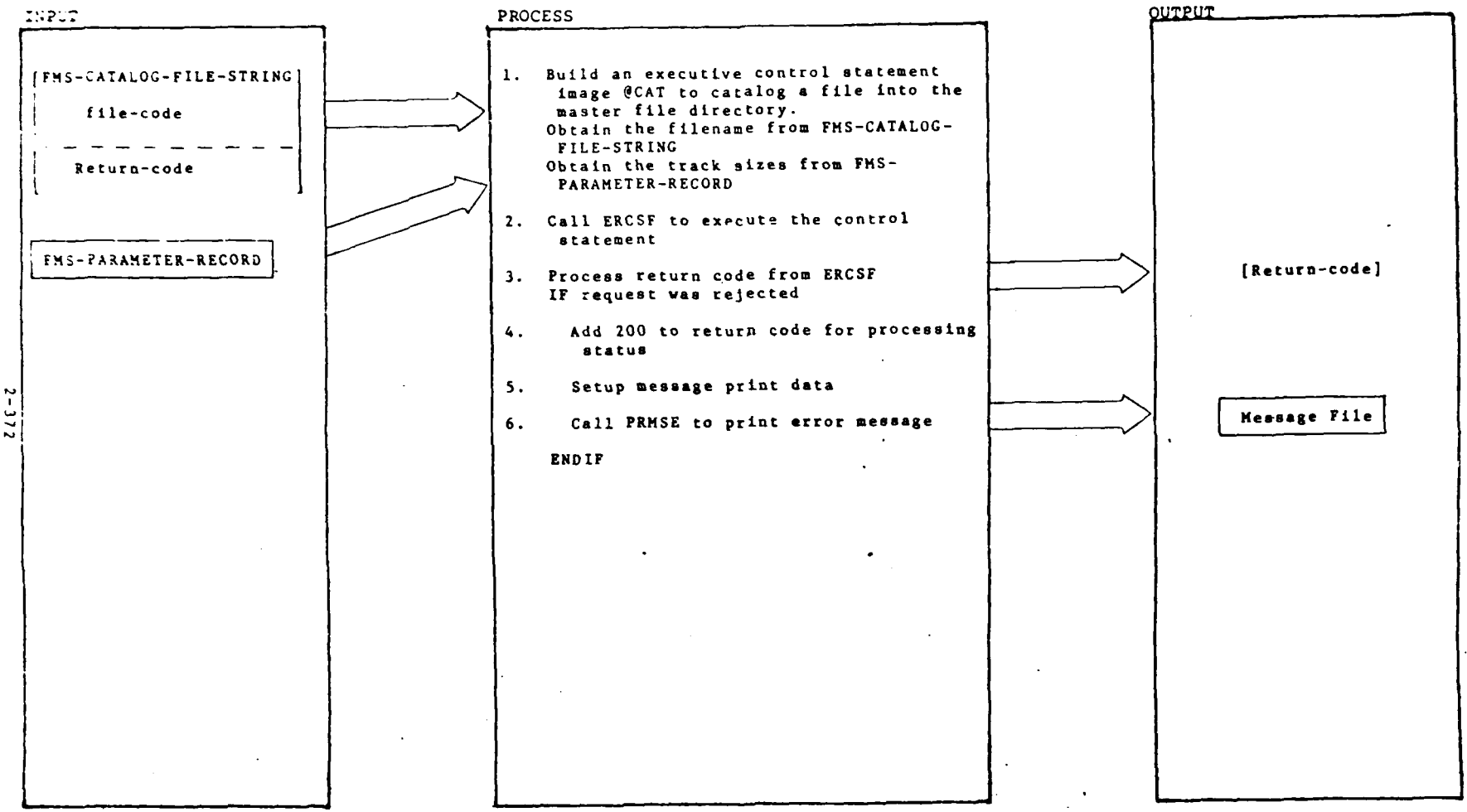


Diagram ID: 1.4.1.4

Name: DAFLE - Deallocate a File

Description: Routine within module ALLOCATOR

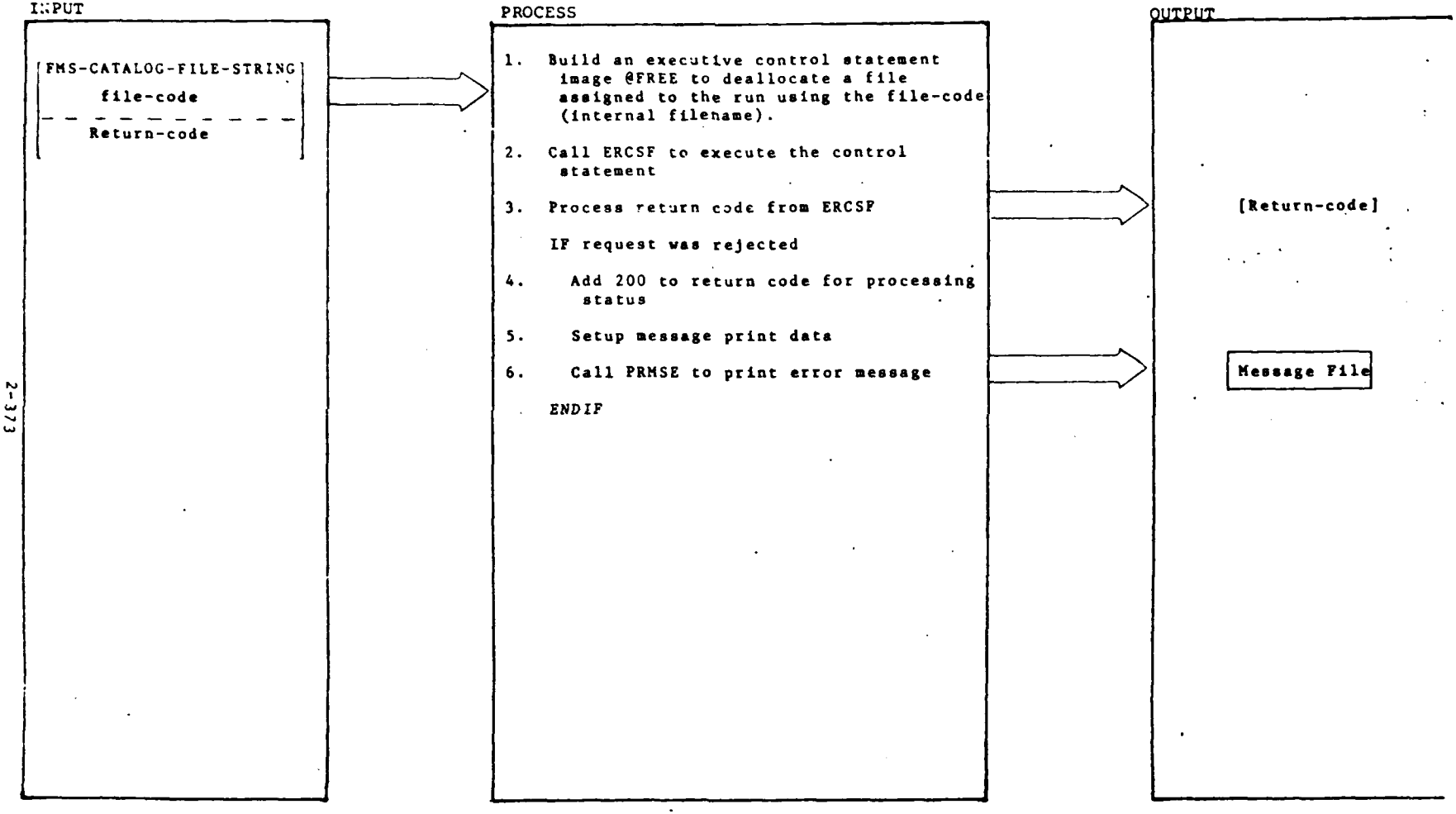


Diagram ID: 1.4.1.5

Name: PGFLE - Uncatalog a File

Description: Routine within module ALLOCATOR

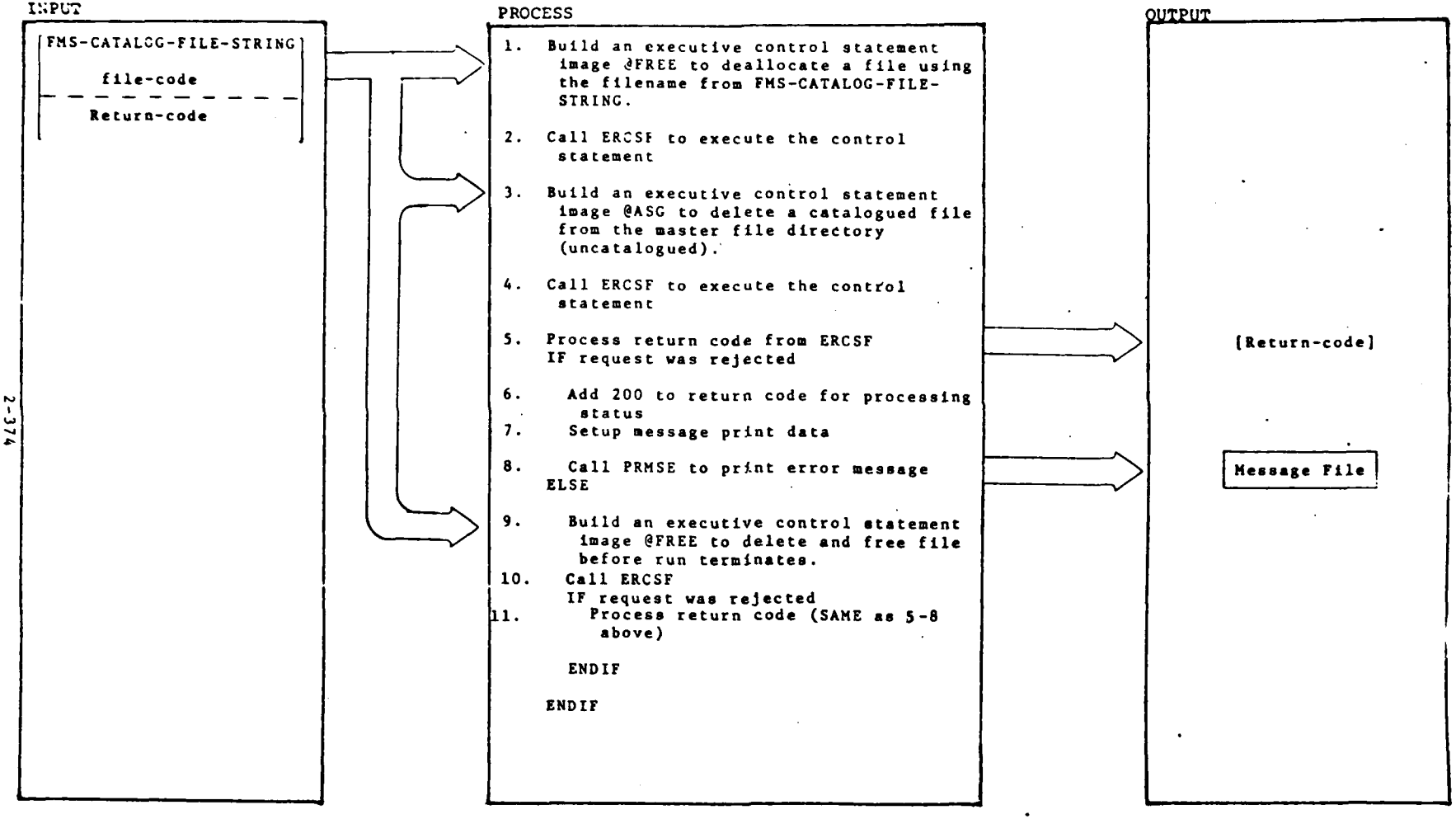


Diagram ID: 1.4.1.6

Name: QYFLE - Query File Information

Description: Routine within module ALLOCATOR

INPUT

FMS-CATALOG-FILE-STRING

file-code

Return-code

PROCESS

1. Build an executive control statement image @USE to attach an internal filename (file code) to the external filename as an alternate reference (obtain filename from FMS-CATALOG-FILE-STRING)

2. Call ERCSF to execute the control statement
   IF request executed successfully

3. Call ERFITEM to obtain file information (number of tracks assigned to file).
   IF request executed successfully

4. Set return code to reserve and maximum track size of file
   ELSE

5. Set return code equal to zero
   ELSE

6. Process return code

7. Add 200 to return code for processing status

8. Setup message print data

9. Call PRMSE to print error message

   ENDIF

OUTPUT

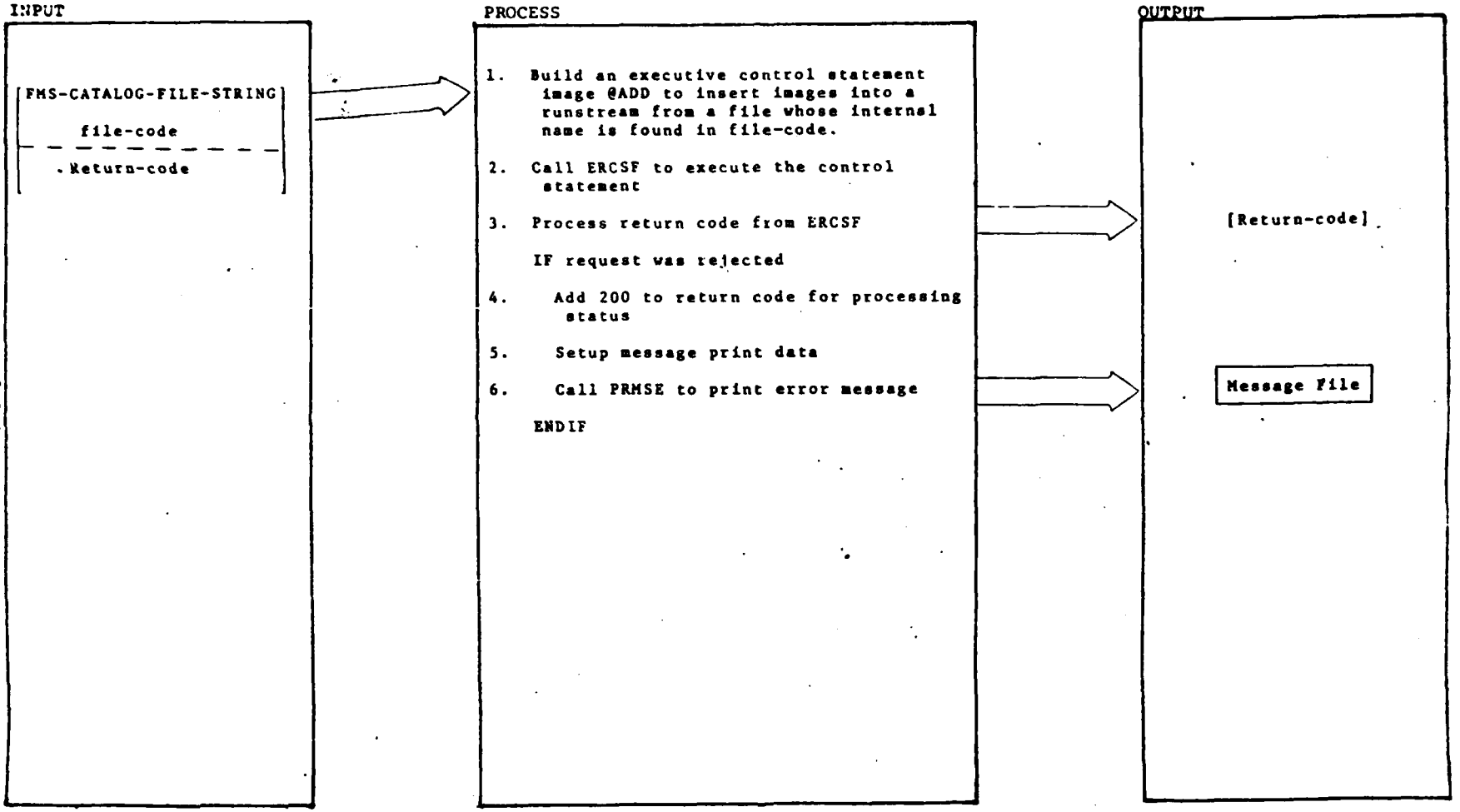
[Return-code]

Message File

Diagram ID: 1.4.1.7

Name: SPJBE - Dynamically expand the runstream

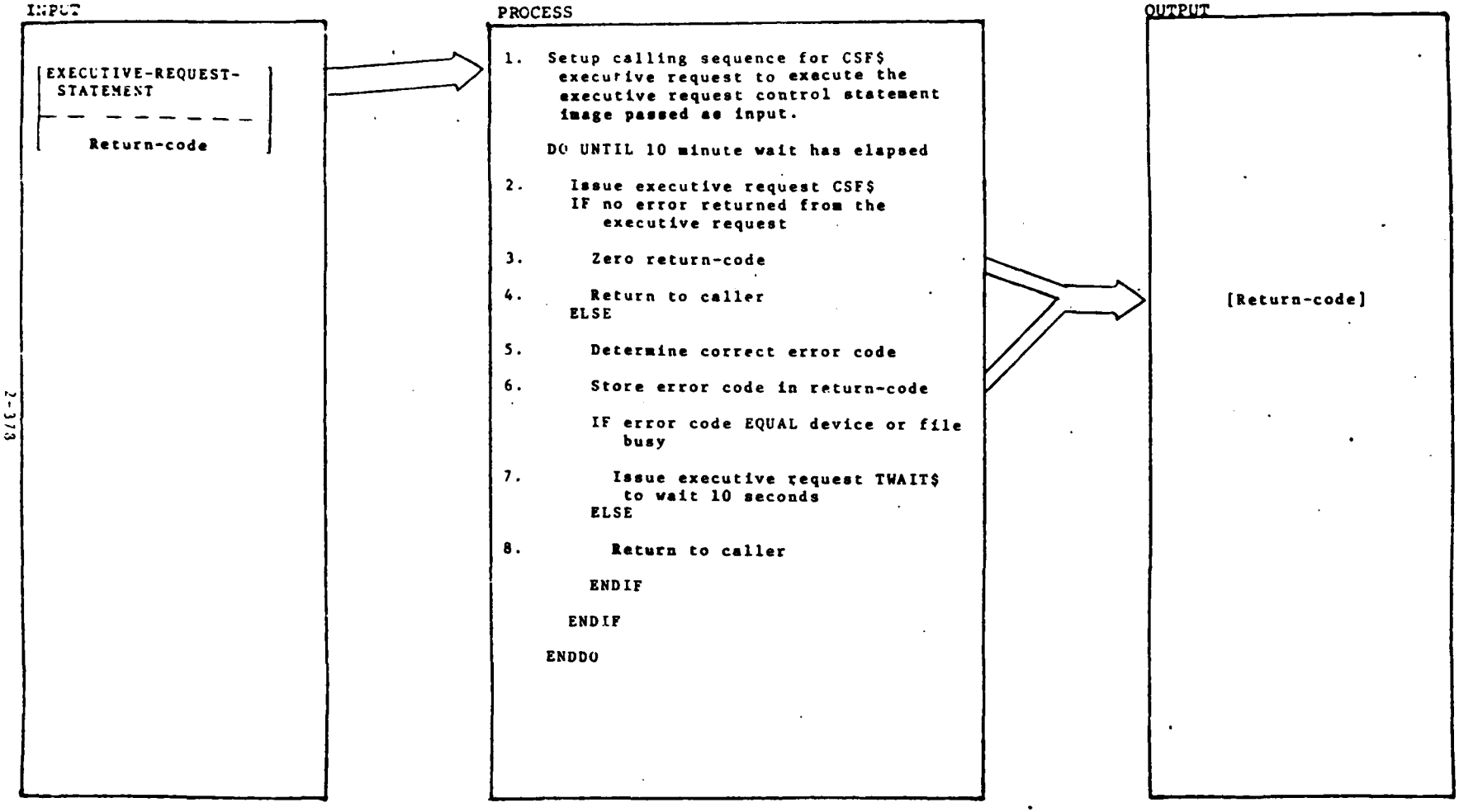
Description: Routine within module ALLOCATOR



2.2.1.4.2 Executive Request Routines

The modules whose operations invoke the Executive Request Functions of the Univac 1100 Operating System are written in Univac 1100 Assembler (Fieldata) language. HIPO diagrams 1.4.2.1 through 1.4.2.3 depict the program operations of each such required module.

Diagram ID: 1.4.2.1 Name: ERCSF Description: Univac Operating System Interface Routine IBM-G-4



IBM-G-4

Diagram ID: 1.4.2.2

Name: ERFITEM (Obtain file information)

Description: Univac Operating System Interface Routine

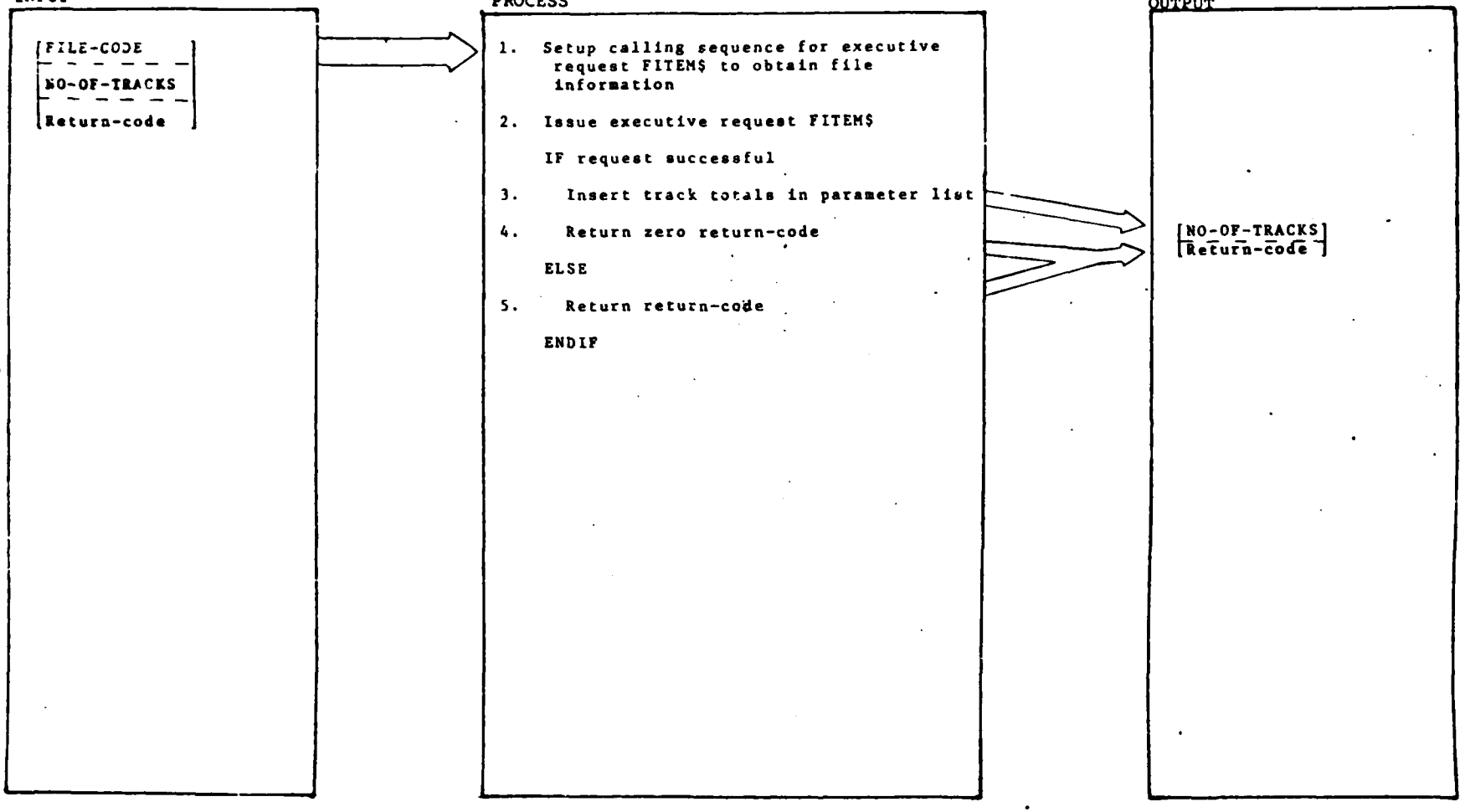
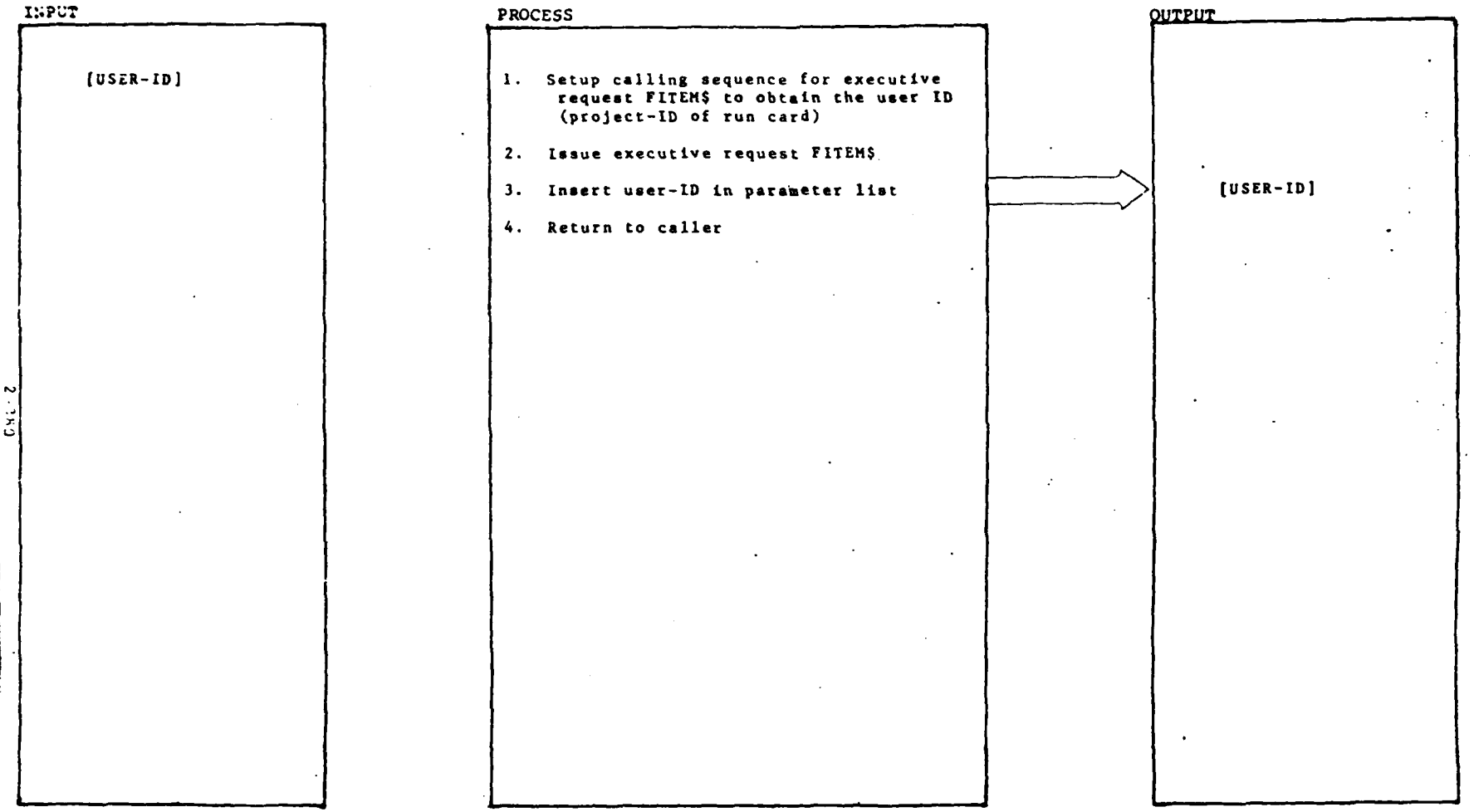


Diagram ID: 1.4.2.3

Name: OBUSE (Obtain User-ID)

Description: Univac Operating System Interface Assembler Routine



SECTION 3. INPUT/OUTPUT DESCRIPTIONS

3.1 General Description

The input to the PSL system consists of directives on input cards (PSL function cards) and data from the user's libraries. A PSL directive which is submitted as a batch-input data card contains the name of the function requested and, if appropriate, sets of keywords and value entries. The user's own program and data cards may be interspersed with the PSL function cards.

The output of the PSL system consists mainly of user data which are stored according to data format in pre-defined files (PSL Sections) and management data reports. The PSL system manipulates the data for each section appropriately for that type of data. The names of a section is one of the following pre-defined section names:

a. SOURCE - source code statements.

b. OBJECT - object module indexes and accounting records.

c. LOAD - load module indexes and accounting records.

d. LINK - collector control cards.

e. JOB - job ocntrol cards used during execution of user program.

f. TEST - test data for use by user program.

g. PDL - program design language statements.

h. TEXT - documentation.

i. MGMT - management data.

j. USER - PSL/non-PSL generated user data.

k. PROGRAM - relocatable and absolute elements.

Management data reports print the contents of one or more management data units. Management data units may contain a combination of automatically collected data and manually input data combined according to the user-defined format unit.

3.2 Characteristics and Description of System Data

3.2.1 PSL Function Cards

Processing within the PSL system is determined by control cards (Function Cards) read by the Batch Control System (BCTLE). As PSL function cards are read, the BCTLE passes control to one of the functional subroutines. When a functional subroutine completes its processing, control is returned to the BCTLE which continues to read PSL function cards and execute functional subroutines until all cards have been processed.

The PSL function card contains the name of the functional subroutine to be called and keyword value parameters, if required. At program execution, the PSL function cards are read and the keyword values specified are stored in the INPUT-CARD-KEYWORD-VALUE table for processing by the called functional subroutine. The INPUT-CARD-KEYWORD-VALUE table (see Section 2.2.1.2.6.2 for description) is built by the Obtain Keyword subfunction (OBKWE) and is made available to other subfunctions of the PSL system.

3.2.2 Types of Data Sections and Contained Data

The PSL project library can support ten different data sets for the storing of user data.

The following paragraphs briefly describe the function and content of each of the project library data sets.

a. SOURCE - Contains the units of source code defined for the different PSL supported languages. It serves as input to the precompilers during compilation of a program.

b. OBJECT - Contains the object module index and accounting records. It serves as locator for object modules stored in PROGRAM section.

c. LINK - Contains the control cards to direct the collector in building the project load modules. The resulting load module is recorded in the LOAD section, and stored in the PROGRAM section.

d. LOAD - Contains the load module index and accounting records. Load modules recorded in this section may be executed by standard job input statements (which are stored in the JOB section) or may be called or invoked by other programs.

e. JOB - Contains job control statements which are used to control execution of programs being developed and other programs used to support the PSL. Data from this section is used to build a job stream for input to the computer.

f. TEST - Contains data used in testing programs under development or maintenance.

g. MGMT - Contains management data related to program development and maintenance. This data is used in the generation of management reports and summaries.

h. TEXT - Contains any textual type information. Data from this file may be used for various types of program or system documentation.

i. PDL - Contains the program design language statements. Data from this file may be used for design, program documentation and system verification.

j. USER - Contains data generated by non-PSL functions.

k. PROGRAM - Contains all object and load modules (relocatable and absolute elements).

A section in a library under the PSL system is created in the format of a PSL Standard File containing 128-word blocks and is catalogued under the User's project. The first block in the file is the Control Block and contains information pertinent to the entire section such as the section name, user specified options, file size, first index block number and space allocation flags. Pointers to the location of each data unit within the section are stored in the index block. The organization is shown in Table 3-1. However, the PROGRAM section is not a PSL Standard File but is a Sperry Univac 1100 series standard Program file.

3.2.3 Management Data

The Management Data Collection and Reporting subsystem of the PSL includes those activities which are related to the acquiring and reporting of management data. A management plan, stored as a unit in the management data section, is used to describe the hierarchical structure of the management data reports. Individual units defined by level contain information which determines the order and source of data items to be collected and summarized.

Data associated with modules may be obtained from two sources; namely, the unit accounting record and by manual input. Manual data may be added, changed or deleted.

Manual inputs are also made to maintain exception checking specifications; i.e., two numeric data items at the same format level may be subjected to a value variance check. The data item values are compared at the time the data is collected and a determination made as to whether an exception exists, pertinent information is added to the data collection unit so that management reports may flag the excepted data item.

The data collection activity specified in the management plan automatically verifies and cross-relates plan level elements, format requirements and data source availability.

Management reports are generated using the collected data. The report format provides identification of the level data and the specific items for which values are printed.

The Management Data Section is composed of units which contain the following types of data.

Management Data Plan Unit - This unit defines the management data report structure.

Management Data Format Unit - The unit defines the data items that may be reported at the following levels:

System level
Subsystem level
Module level
Job level
Unit level

PROJECT DESCRIPTION
PROJECT START DATE
ESTIMATED COMPLETION DATE
ACTUAL COMPLETION DATE
PLANNED AVERAGE YEARS EXPERIENCE MANAGERS
PLANNED AVERAGE YEARS EXPERIENCE ANALYSTS
PLANNED AVERAGE YEARS EXPERIENCE PROGRAMMERS
PLANNED AVERAGE YEARS EXPERIENCE ADMINISTRATIVE
PLANNED AVERAGE YEARS EXPERIENCE OTHER
ACTUAL AVERAGE YEARS EXPERIENCE MANAGERS
ACTUAL AVERAGE YEARS EXPERIENCE ANALYSTS
ACTUAL AVERAGE YEARS EXPERIENCE PROGRAMMERS
ACTUAL AVERAGE YEARS EXPERIENCE ADMINISTRATIVE
ACTUAL AVERAGE YEARS EXPERIENCE OTHER
PLANNED NUMBER OF MANAGERS
PLANNED NUMBER OF ANALYSTS
PLANNED NUMBER OF PROGRAMMERS
PLANNED NUMBER OF ADMINISTRATIVE
PLANNED NUMBER OF OTHER
ACTUAL NUMBER OF MANAGERS
ACTUAL NUMBER OF ANALYSTS
ACTUAL NUMBER OF PROGRAMMERS
ACTUAL NUMBER OF ADMINISTRATIVE
ACTUAL NUMBER OF OTHER
ESTIMATED PERSONNEL TURNOVER RATE
ACTUAL PERSONNEL TURNOVER RATE
ESTIMATED LOCAL TRAVEL
ACTUAL LOCAL TRAVEL
ESTIMATED DISTANT TRAVEL
ACTUAL DISTANT TRAVEL
ESTIMATED WORKING CONDITIONS
ACTUAL WORKING CONDITIONS
PLANNED PROGRAMMING LANGUAGE EXPERIENCE
ACTUAL PROGRAMMING LANGUAGE EXPERIENCE
PLANNED SIMILAR APPLICATION EXPERIENCE
ACTUAL SIMILAR APPLICATION EXPERIENCE
PLANNED TARGET COMPUTER EXPERIENCE
ACTUAL TARGET COMPUTER EXPERIENCE
ESTIMATED CUSTOMER APPLICATION EXPERIENCE
ACTUAL CUSTOMER APPLICATION EXPERIENCE
ESIMTATED CUSTOMER EQUIPMENT EXPERIENCE
ACTUAL CUSTOMER EQUIPMENT EXPERIENCE
ESTIMATED COMPLEXITY OF PROJECT
ACTUAL COMPLEXITY OF PROJECT

Table 3-2. System Level Management Data Input

ESTIMATED PAGES DOCUMENTATION FUNCTIONAL SPECS
ESTIMATED PAGES DOCUMENTATION USER GUIDE
ESTIMATED PAGES DOCUMENTATION TEST SPECIFICATIONS
ESTIMATED PAGES DOCUMENTATION PROGRAM LISTINGS
ACTUAL PAGES DOCUMENTATION FUNCTIONAL SPECS
ACTUAL PAGES DOCUMENTATION USER GUIDE
ACTUAL PAGES DOCUMENTATION TEST SPECIFICATIONS
ACTUAL PAGES DOCUMENTATION PROGRAM LISTINGS
SYSTEM NAME
SUBSYSTEM NAMES
NUMBER SUBSYSTEM LEVELS
NUMBER PROGRAMS FOR SUBSYSTEMS
ESTIMATED NUMBER OF NON-OVERLAPPING FIELDS
ACTUAL NUMBER OF NON-OVERLAPPING FIELDS
ESTIMATED NUMBER INPUT FORMATS
ACTUAL NUMBER INPUT FORMATS
ESTIMATED AVERAGE FIELDS FOR INPUT FORMATS
ACTUAL AVERAGE FIELDS FOR INPUT FORMATS
ESTIMATED NUMBER OUTPUT FORMATS
ACTUAL NUMBER OUTPUT FORMATS
ESTIMATED AVERAGE FIELDS FOR OUTPUT FORMATS
ACTUAL AVERAGE FIELDS FOR OUTPUT FORMATS
NUMBER SYSTEM TEST RUNS PLANNED
NUMBER SUCCESSFUL SYSTEM TEST RUNS
NUMBER SYSTEM TEST RUNS EXECUTED
ITEM TO BE DELETED DURING TESTS
LINES OF SOURCE CODE INPUT
NUMBER OF COMPILATIONS
NUMBER OF LINES IN CYCLE
NUMBER OF UPDATES IN CYCLE
NUMBER OF UPDATES
NUMBER OF UNITS
AVERAGE UNIT SIZE
MAXIMUM UNIT SIZE
NUMBER OF MODULES

Table 3-2. System Level Management Data Input (Cont.)

SUBSYSTEM NAME
LINES OF SOURCE CODE INPUT
NUMBER OF MODULES
NUMBER OF UNITS
AVERAGE UNIT SIZE
MAXIMUM UNIT SIZE
AVERAGE MODULE SIZE
MAXIMUM MODULE SIZE
NUMBER OF LINES IN CYCLE
NUMBER OF UPDATES IN CYCLE
NUMBER OF UPDATES
AVERAGE NUMBER OF UPDATES
MAXIMUM NUMBER OF UPDATES
CURRENT CYCLE DATE
PREVIOUS CYCLE DATE
MAN-MONTHS OF EFFORT BUDGET
MAN-MONTHS OF EFFORT EXPENDED
MATERIAL COSTS BUDGET
MATERIAL COSTS EXPENDED
PERSONNEL COSTS BUDGET
PERSONNEL COSTS EXPENDED
TRAVEL COSTS BUDGET
TRAVEL COSTS EXPENDED
COMPUTER TIME BUDGET
COMPUTER TIME EXPENDED
COMPUTER COSTS BUDGET
COMPUTER COSTS EXPENDED
MISCELLANEOUS COST BUDGET
MISCELLANEOUS COST EXPENDED

Table 3-3. Subsystem Level Management Data Input

MODULE NAME
ORIGINATING PROGRAMMER
ORIGINATING START DATE
LAST MODIFIED DATE
PROGRAM LANGUAGE
LINES OF SOURCE CODE INPUT
NUMBER OF UNITS
AVERAGE UNIT SIZE
MAXIMUM UNIT SIZE
NUMBER OF LINES ADDED
NUMBER OF LINES DELETED
NUMBER OF LINES CHANGED
NUMBER OF COMPILATIONS
NUMBER OF LINES IN CYCLE
AVERAGE UPDATE CYCLE
NUMBER OF UPDATES
AVERAGE NUMBER OF UPDATES
MAXIMUM NUMBER OF UPDATES
CURRENT CYCLE DATE
PREVIOUS CYCLE DATE
LINES OF SOURCE CODE COPIED
REAL UNIT COUNT
STUB UNIT COUNT
STRUCTURED PROGRAM ERROR COUNT
MODULE TYPE

Table 3-4. Module Level Management Data Input

UNIT TYPE
LINE SIZE
INCLUDE UNIT NAME
VERSION
MODIFICATION
DATE ORIGINATED
ORIGINATORS NAME
DATE OF LAST UPDATE
TIME OF LAST UPDATE
UNIT LANGUAGE
INCLUDE COUNT
LINES IN UNIT
STRUCTURED PROGRAM ERROR SWITCH
LINES ADDED TO UNIT
LINES CHANGED IN UNIT
LINES DELETED IN UNIT
LINES INPUT TOTAL
LINES INPUT CYCLE
LINES COPIED IN UNIT
UPDATES TOTAL
UPDATES CYCLE
USER WORK AREA

Table 3-5. Unit Level Management Data Input

PROGRAM IDENTIFICATION
JOB NAME
NUMBER OF RUNS
NUMBER OF RUNS FOR CYCLE
CURRENT CYCLE DATE
PREVIOUS CYCLE DATE
AVERAGE TURNAROUND TIME PER JOB

Table 3-6. Job Level Management Data Input

The system level data unit contains a description of the system parameters and a list of the subsystems which comprise the system. See Table 3-2 for a list of valid system level parameters.

The subsystem, module and unit level data formats contain a description of the parameters unique to each level. A description of valid parameters for each level is contained in Tables 3-3 through 3-6.

Management Data Input Units - This unit contains inserted data gathered at the following levels:

a. System description data.

b. Subsystem description data.

c. Module description data.

d. Job information.

Data records, as manually created for these data levels, are placed in the Management Data File. Data may be added, changed or deleted from an existing unit in the file.

Management Data Collection Units - Management data collection is supported at two different levels. First, at the unit level, management data is automatically collected and added to an expanded Unit Accounting record. This collection is initiated for project units as they are built with the management data option. The second level of management data describes a system, subsystem, module or JOB. This data is manually input to the Management Data file for the project. See Table 3-7 for a description of the Management Data Base Structure.

3.3 PSL Copy Library

The copylib unit contains a collection of data items that are used by various PSL functions within the system. The copy library definitions reside in the Source Section of the library under the unit name, CPYLIB.

The data components within the PSL system can be categorized as being one of the following:

Applicable PSL Functions	* * * * * PSL MGMT Section * * * * *
	Management Data Plan Unit (MDCR-PLAN)
	SYSTEM = BIGTOP
	JOB = BIGJOB
	SUBSYS=SUB1, MODULE=MOD1A, MODULE=MOD1B
	SUBSYS=SUB2, MODULE=MOD2A, MODULE=MOD2B
	Management Data Format Units (MDCR-FORMAT-)
MDFORMAT	SYSTEM SUBSYS MODULE JOB UNIT
	Management Data Input Units
MDUPDATE	BIGTOP BIGJOB SUB1 SUB2 MODIA ETC.
MDXCHECK	
	Management Data Collection Units
MDCOLLECT	MDCR-COLLECTION MDCR-ARCHIVE-001-(DATE1)
PURGE	MDCR-ARCHIVE-002-(DATE2)
	MDCR-ARCHIVE-003-(DATE3)

* * * * * PSL SOURCE Section(s) * * * * *

	Top Units	Included Units
ADD		
CHANGE	MOD1A	
MOVE	MOD1B	
REPLACE	MOD2A	
PURGE	MOD2B	

a. data used or generated by PSL function

b. data used to control the execution flow of the system.

c. data recorded as a result of systm monitoring.

d. data used to generate management reports.

The data items in CPYLIB are grouped according to their usage in the system and are presented alphabetically by group name. The various groups and a description of usage of each group is presented below. A more detailed description of individual data items is presented in Section 2 of this manual under Data File and Table Descriptions associated with each function processor.

Group Label	Usage Description
ACCESS-MODE	Defines the types of permissible accesses to a PSL file.
ACCESS-MODE-VALUE	Pre-defined assignments for sequential and random accessing.
ACCOUNTING-INFORMATION	Data associated with each unit, maintained in the accounting record and which reflects all activity performed on that unit. Data is entered automatically at the completion of each activity.
ADUN-LINK-NAMES	Collector defined program modules processing.
ALLOCATION-TYPE	Valid allocation types that may be specifying when referencing a file.
ALLOCATION-TYPE-VALUE	Pre-defined assignments for each type of allocation.
BCTL-LINK-NAMES	Collector defined program modules which are used in the Batch Control processing.
CHANGE-RECORD	Parameters used by the CHANGE function when modifying the contents of a unit.

Group Label	Usage Description
CHUN-LINK-NAMES	Collector defined program modules which are used in the CHANGE UNIT processing.
CODES-FOR-ACCESS-METHODS	Error codes for any failures encountered while accessing a PSL block.
CODES-FOR-ADUN	Error codes encountered by the ADD-A-UNIT function.
CODES-FOR-BCTL	Error codes encountered by the main control module.
CODES-FOR-BKLB	Error codes encountered by the BACK-UP LIBRARY function.
CODES-FOR-BKSC	Error codes encountered by the BACK-UP SECTION function.
CODES-FOR-CHUN	Error codes encountered by the CHANGE UNIT function.
CODES-FOR-CMML	Error codes encountered by the COMPILE function.
CODES-FOR-CRSE	Error codes encountered by the CREATE SECTION function.
CODES-FOR-DLUN	Error codes encountered by the DELETE UNIT function.
CODES-FOR-ESPA	Error codes encountered by the ESTABLISH PARAMETERS function.
CODES-FOR-EXPG	Error codes encountered by the EXECUTE PROGRAM function.
CODES-FOR-FMS-SETUP	Error codes encountered by the file management parameter processor.
CODES-FOR-INDEX-PROCESSING	Error codes encountered by the index processing module.
CODES-FOR-ITPJ	Error codes encountered by the INITIALIZE PROJECT function.

Group Label	Usage Description
CODES-FOR-LKPG	Error codes encountered by the LINK PROGRAM function.
CODES-FOR-MVUN	Error codes encountered by the MOVE UNIT function.
CODES-FOR-PCLU	Error codes encountered by the PROCESS LOWER UNIT function.
CODES-FOR-PGUN	Error codes encountered by the PURGE UNIT function.
CODES-FOR-PRAU	Error codes encountered by the PRINT AUTHOR function.
CODES-FOR-PRCB	Error codes encountered by the PRINT COBOL function.
CODES-FOR-PRDC	Error codes encountered by the PRINT DOCUMENT function.
CODES-FOR-PREPROCESS-LINK	Error codes encountered by the PREPROCESS LINK function.
CODES-FOR-PRMR	Error codes encountered by the PRINT MANAGEMENT REPORT function.
CODES-FOR-PRSD	Error codes encountered by the PRINT SOURCE DATA function.
CODES-FOR-PRUN	Error codes encountered by the PRINT UNIT function.
CODES-FOR-READ-WRITE-UNIT	Error codes encountered during reading/writing of a unit.
CODES-FOR-RPUN	Error codes encountered by the REPLACE UNIT function.
CODES-FOR-RSLB	Error codes encountered by the RESTORE LIBRARY function.
CODES-FOR-SP-CHECK	Error codes encountered by the structured code check processor.
CODES-FOR-STRUCTURES	Status and error codes used during processing of structured figures.

Group Label	Usage Description
CODES-FOR-PGSC	Error codes encountered by the PURGE SECTION function.
CODES-FOR-WRAC	Error codes encountered during update of accounting record.
CODES-FOR-WRJB	Error codes encountered during control card processing.
CONDITION-SETTINGS	Pre-defined assignments for TRUE/FALSE switches.
CONTROL-CARD	Formats of the various control cards generated by PSL.
DEFAULT-LINES-PER-UNIT	Default value for maximum number of lines per unit.
FILE-CONTROL-FOR-ACCESS	Names each file used in the non-spawn environment of the system.
FILE-NAMES	File names used by the PSL system when initializing a System/ Project file.
FILE-SIZES	Initial space assignment values for each type of section file.
FMS-CATALOG-FILE-STRING	Literal used for specifying a fully qualified FMS file name.
FMS-VOLINFO	Device specification for a file.
HEADER-FOR-UNIT-LISTING	Report title line for PRINT UNIT function.
INDEX-ENTRY	Format of entry in index block for each unit maintained by PSL.
INITIAL-PARAMETER	User Project - Identification Number.
INPUT-CARD	Workarea for storage of input data cards.

Group Label	Usage Description
INPUT-CARD-KEYWORD-VALUE	Describes valid parameters for the input card.
ITPJ-LINK-NAMES	Collector defined program modules which are used in the INITIALIZE PROJECT processing.
JCL-CARD	Workarea for storage of input control cards by ** JCL function.
JOB-CARD	Describes valid formats for PSL control cards.
KEYWORD-CARD	Cards containing keyword values used in the spawn job environment.
LANGUAGE-LINK-TABLE	Print program modules used by the PSL language processor.
LANGUAGE-TABLE	Not currently used.
LENGTH-OF-KEYWORD-VALUE	Maximum length specifications for standardized keywords.
LINK-NAMES-FOR-BKLB	Collector defined program modules which are used in the BACKUP LIBRARY processing.
LINK-NAMES-FOR-BKSC	Collector defined program modules which are used in the BACKUP SECTION processing.

Group Label	Usage Description
LINK-NAMES-FOR-CMML	Collector defined program modules which are used in the COMPILE processing.
CRSC-LINK-NAME	Collector defined program modules which are used in the CREATE SECTION processing.
LINK-NAMES-FOR-DLUN	Collector defined program modules which are used in the DELETE UNIT processing.

Group Label	Usage Description
LINK-NAMES-FOR-ITSF	Collector defined program modules which are used in the INITIALIZE STANDARD FILE processing.
LINK-NAMES-FOR-LKPG	Collector defined program modules which are used in the LINK PROGRAM processing.
LINK-NAMES-FOR-MVUN	Collector defined program modules which are used in the MOVE UNIT processing.
LINK-NAMES-FOR-PGSC	Collector defined program modules which are used in the PURGE SECTION processing.
LINK-NAMES-FOR-PRAU	Collector defined program modules which are used in the PRINT AUTHOR processing.
LINK-NAMES-FOR-PGUN	Collector defined program modules which are used in the PURGE UNIT processing.
LINK-NAMES-FOR-PRMS	Collector defined program modules which are used in the PRINT MESSAGE processing.
LINK-NAMES-FOR-RSLB	Collector defined program modules which are used in the RESTORE LIBRARY processing.
MSG-DATA-FOR-ITSF	Error messages data area for FMS accesses.
MESSAGE-LINE	Workarea for output message processing.
MSG-NBRS-FOR-ITSF	Error codes produced by the INITIALIZE STANDARD FILE function.
MVUN-UNIT-TYPE-VALUES	Unit type designators for receiving/sending units processed by MOVE UNIT function. Related to UNIT-TYPE-TABLE.

Group Label	Usage Description
CODES-FOR-OBIC	Pre-defined status/error codes produced by input card processing.
MSG-NBRS-FOR-BCTL	Pre-defined status/error codes produced by the main control unit.
PARAMETER-TABLE	Used to store PSL parameters that may be applied to several PSL functions.
PRSD-LINK-NAMES	Collector defined program modules which are used in the PRINT SOURCE DATA processing.
PRXX-LINK-NAMES	Collector defined program modules which are used in the PRINT INDEX processing.
PSL-CARD-IDENTIFIER	Prefix for columns 1-3 on PSL function input cards.
PSL-CONTROL-BLOCK	Contains information pertinent to the entire PSL section, such as section name, user-selected options, file size, and the block flags used to control the space allotment within the file.
PSL-DATA-BLOCK	Contains PSL unit data, unit accounting information, and unit control information.
PSL-INDEX-BLOCK	Contains entries pointing to the location of each data unit within the section.
RANDOM-FD	File description of a random PSL file giving the physical structure identification.
PCLU-PROCESS-TYPE-VALUE	Values which denotes the addition/ deletion of an included unit during processing of a unit.

Group Label	Usage Description
PRINT-CLASSIFICATION	Default print classification.
PRINT-UNIT-OPTION-DEFAULTS	Default print values for listing of units.
PROC-STATUS-FOR-ACCESS-METHODS	Status/error codes returned on file access.
PROJECT-INDEX-ENTRY	Format of entries in the project index block.
PROJECT-DIRECTORY-ENTRY	Format of entries in the project directory block.
RANDOM-RECORD	PSL random record storage area.
REQUESTED-INDEX-ENTRY	Workarea used to store parameters needed to access a unit.
SCSG-LINK-NAME	Collector defined program modules which are used by the SCAN STRING function.
SECTION-OPTIONS	Table containing selected section options which are appropriate for each section type.
SECTION-TABLE	Table of valid section types and their associated section codes supported by PSL.
STCF-STMF-LINK-NAMES	GMAP program names used by the loader in the SET UP TO CREATE/MODIFY FILE function.
SUPPORT-LANGUAGE-TABLE	Table of languages supported by PSL.
UNIT-CARD-RECORD	Unit input record description.
UNIT-LISTING-LINE	Unit output line description 132 character print line.
UNIT-TAPE-RECORD	Unit tape output record description.

Group Label	Usage Description
UNIT-TYPE-TABLE	Defines types of valid units processed by the PSL.
UNIT-TYPE-WORDS	Unit types name descriptors.
USER-TYPE-MASTER	Defines the user type of the first project directory entry.

SECTION 4. PROGRAM ASSEMBLING, LOADING AND MAINTENANCE PROCEDURES

4.1 Programming Support Library (PSL) Structure

The entire PSL system is contained in one program file (PSLPRG). The PSLPRG program file is composed of symbolic, relocatable and absolute elements, which define the PSL system. The PSL system is functionally divided into operational elements and maintenance elements. The following naming convention is implemented to clarify which functional area is being addressed.

1. The qualifier "PSL." is used to reference the operational elements.
2. The qualifier "S." is used to reference the maintenance elements.

The PSLPRG program file may be saved on tape with standard UNIVAC backup utility. The program file may be restored when needed for system maintenance. A listing of the element table for PSLPRG is shown in Appendix C.

4.1.1 General Information

The symbolic elements of the PSLPRG program file contain source code, CPYLIB (common source code), collection specifications, test data, installation procedures and general procedures. The relocatable elements contain the compiled (assembled) source code for the various PSL functions. The absolute elements contain collections of various relocatable elements needed for program execution and PSL processing. The operational PSL system is composed of seven absolute elements (PRMD, PRPS, CLMD, BCTL, PPCB, PPFT, PPGL) and one general procedure (RUN). By convention, "PSL." is the qualifier for the absolute elements and the project-id is the qualifier for the procedure "RUN".

4.1.1.1 Symbolic Elements

The symbolic elements in the PSLPRG program file have the functions as outlined in Sections 4.1.1.1.1 - 4.1.1.1.4

4.1.1.1.1 Maps[1]

The elements BCTLMAP[2],PRPSMAP,PPFTMAP,PRMDMAP,PPGLMAP,CLMDMAP and PRECBMAP specify what relocatable elements are to be collected for the various PSL functions. The absolute elements which these elements generate are BCTL, PRPS,PPFT,PRMD,PPGL,CLMD and PRECOMPILER, respectively. The main program is the first relocatable element named in the map. The map elements collect modules for the following PSL processing.

a BCTLMAP - Batch control processing
b PRPSMAP - Print program structure processing
c PPFTMAP - Structured FORTRAN processing
d PRMDMAP - Print MANAGEMENT Data processing
e PPGLMAP - General preprocessor processing
f CLMDMAP - Collect MANAGEMENT Data processing
g PRECBMAP[3] - Structured COBOL processing

The following example shows an absolute module generation:

```
@USE PSL.,PSLPRG.
@PREP PSL.
@MAP PSL.BCTLMAP,.BCTL
```

4.1.1.1.2 COBOL Procedure Library

The element CPYLIB is used to produce the PROC element PROCLIB, which allows common source code to be accessed by different programs via the COBOL 'COPY' statement. PROCLIB is generated by the @PDP control card processing. The following example shows the generation of PROCLIB.

```
@USE S.,PSLPRG.
@USE COB$PF.,S.
@PDP,CW S.CPYLIB,.PROCLIB
```

Note

[1] refer to appendix B for listing of elements
[2] BCTL segmented for efficient use of storage
[3] stand alone COBOL precompiler

4.1.1.1.3 Source Data

The elements which represent the source code of the PSL system programs. For a detailed description of each program, reference section 2.2. Modification of source code may be performed by utilizing the UNIVAC 1100 series editing facilities.

4.1.1.1.4 Test Data

There are three sets of test data in the PSLPRG program file. The PSL System test data is stored in elements 1A through 8B. The PSL Supplemental test data is stored in elements S1 through S3. The PSL installation test data is stored in element PSL-INSTTEST.

The following example shows the PSL batch processing being evoked using a test data element.

```
@USE PSL.,PSLPRG.
@USE S.,PSLPRG.
@ADD S.RUN
@ADD,D S.1A
@END PSL
@XQT PSL.BCTL
@EOF
```

4.2 PSL Procedures

The PSL JOB section contains the JCL procedures needed to invoke PSL programs. The JCL procedures can be classified as either language dependent or general. A brief description of JCL procedures under each classification follows. Figure 4-03 shows a card deck setup to add a new procedure.

```
Col     Col     Col
1       8       16

$       IDENT   . . .
$       USERID  new-umc-name$new-umc-password/print-classification
$       SELECT  new-umc-name/CJPSL/B
** PARAM PROJECT=new-umc-name,LIBRARY=PSL,SECTION=JOB,UTYPE=MAIN
** ADD   UNIT=new-procedure-name

        (USER's procedure cards go here.)

$        ENDCOPY CC
$        ENDJOB
```

Note: Maximum length of procedure name is eight characters.
Control cards "@ADD,D" should be altered for ADD processing and later corrected by CHANGE processing.

Figure 4-03. Adding a New Procedure

4.1.1.2 Relocatable Element

The assembled PSL source programs are stored as relocatable elements bearing the same name as their related symbolic elements. The relocatable elements can be generated by PSL procedures[1], the UNIVAC ASSEMBLER or the PSL COBOL stand-alone precompiler. A sample card deck which compiles a PSL program using the stand-alone precompiler is shown in figure 4-02.

4.1.1.3 Absolute Elements

The absolute elements are the load modules needed to execute the PSL system. PSL load modules are produced by the standard UNIVAC 1100 series collection processor. Refer to section 4.1.1.1.1 for a description of the contents of each absolute element.

Note

[1] refer to section 4.2

```
@USE PSL.,PSLPRC.
@USE S.,PSLPRG.
@ASG,T PI,F///100
@ASG,T PO,F///100
@ASG,T PR,F///100
@ASG,T LS,F///100
@DATA,I PI.
@ADD,D S.BCTLE  (SOURCE CODE)
@END
@XQT S.PRECOMPILER  (PRECOMPILER)
@ACOB,ITVSEC .,S.BCTLE (RELOCATEABLE)
@ADD PO.
@EOF
```

Figure 4-02 Sample Run For Compiling PSL Program

4.2.1 General JCL Procedures

The general procedures are stored in the following units of the JOB section with the exception of "RUN".

a. Run - Invokes the Batch PSL system. (All PSL functions are initially processed by this procedure).

b. CLMD - Collects management data.

c. MD - Prints Management Data report.

d. PS - Prints Program Structure report.

4.2.2 Language Dependent Procedures

The language dependent procedures enable the user to precompile and assemble the PSL supported languages. Job procedures are provided for the structuring, as well as, the INCLUDE processing of the supported languages.

a. COBOL - Compiles COBOL code without using precompiler.

b. SCOBOL - Processes COBOL code with precompiler and compiles resulting code.

c. COBOLG - Processes COBOL code with INCLUDE capability and compiles resulting code.

d. PSLCOB - Processes COBOL code with precompiler, PSL CPYLIB and compiles resulting code.

e. FORTRAN - Compiles FORTRAN code without using precompiler.

f. ASM - Compiles ASSEMBLY code with ASSEMBLY LANGUAGE compiler without using precompiler.

g. FORTRANG - Processes FORTRAN code with INCLUDE capability and compiles resulting code.

h. SPFORT - Processes FORTRAN code with precompiler and compiles resulting code.

i. ASMG - Processes ASSEMBLY code with INCLUDE capability and compiles resulting code.

Each of the procedures, except RUN, is used in a spawned job and is modified by the PSL system to reference appropriate file names before the procedure is spawned.

---

[1]Refer to Appendix D of the PSL User's Manual for listings of PSL procedures.

When adding a user procedure special consideration must be made if the procedure contains a "@ADD,D" control card. This control card must be altered so as not to appear as directives in the PSL run stream. Once the procedure has been added, the **CHANGE function can be used to reformat these cards.

4.2.2.1 Compile Procedures

Language dependent procedures may be invoked by the keyword PROCEDURE on the **COMPILE function card or by referencing the language name of the unit being compiled. The keyword PROCEDURE will override the procedure name which would have been invoked by the language name.

When a compile procedure is added, the PSL system requires certain flag-words in columns 73 through 80 (left-justified). An example of a compile procedure with the flag-words is shown in Figure 4-04.

Card A is any control card which will invoke the compiler. The card must have the word "COMPILE$" in columns 73 through 80. The user may set the options on this card, otherwise the default options will apply. If a user ** COMPILE function requests OBJECT=NO, no permanent copy of the relocatable is saved.

Card B defines the element in which the compiler expects to find the source input. This card must have the word "SOURCE$$" in columns 73 through 80. The PSL COMPILE processor will substitute for card B an @ADD,D card containing the same element code as card B. This @ADD,D card will contain the catalog-file-string of the independent PSL file in the SOURCE section of the user's PSL library which contains the source code.

When the COMPILE function is directed to save the object output (OBJECT=YES), the PSL COMPILE processor will substitute the user's program file for "TPF$" on card A.

Card	Col 1	Col 73
A	@FOR,SWT	COMPILE
B	@ADD	SOURCE
C	@EOF	

(other cards as required)

Figure 4-04 COMPILE Prodecure

4.3 MANAGEMENT Section

The PSL MANAGEMENT section is created and maintained for each project for which management data is collected. The units of the MANAGEMENT section of library PSL are available to aid the user. The units presented gives the user a basic capability to define the store elements of a project for which management data is to be collected. A detailed description of the various units in the MANAGEMENT section is given in Section 3.1.6 of the PSL User's Manuel.

APPENDIX A. PROGRAM STRUCTURE MAPS

The following program structure maps delineate the hierarchical organization of structured program modules in the PSL system. The hierarchical level at which a module is "called" or a unit of code is "included" is noted and the name of the called or included code is correspondingly indented.

SEGMENT NAME: ADUNE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	ADUN	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
3	INPUT-OUTPUT SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	LLINK	CALLED
3	STORE-INPUT-KEYWORDS	
4	OBKWE	CALLED
4	PRMSE	CALLED
4	INTERPRET-KEYWORD-VALUES	
5	OBKWE	CALLED
4	PRMSE	CALLED
4	EDIT-INPUT-VALUES	
5	PRMSE	CALLED
5	PRMSE	CALLED
5	PRMSE	CALLED
5	PRMSE	CALLED
5	PRMSE	CALLED
3	ASFLE	CALLED
3	DETERMINE-UNIT-TYPE	
4	FDXEE	CALLED
4	DETERMINE-OLD-UNIT-TYPE	
5	PRMSE	CALLED
5	PRMSE	CALLED
4	DETERMINE-NEW-UNIT-TYPE	
3	EDIT-UNIT-NAME-AND-LANG	
4	PRMSE	CALLED
4	PRMSE	CALLED
4	PRMSE	CALLED
3	SET-UP-ACCOUNTING-INFO	

SEGMENT NAME: ADUNE

UNIT LEVEL	UNIT NAME	UNIT TYPE
4	ITRDE	CALLED
4	TMRDE	CALLED
3	PUT-UNIT-DATA-IN-SECTION	
4	ITWRE	CALLED
4	LLINK	CALLED
4	OBSDE	CALLED
4	STORE-LOWER-UNIT-INFO	
5	PCLUE	CALLED
5	PRMSE	CALLED
4	WRLNE	CALLED
4	PCLUE	CALLED
4	OBSDE	CALLED
4	TMWRE	CALLED
3	CONCLUE-ADDITION-OF-UNIT	
4	WRACE	CALLED
4	CHXEE	CALLED
4	ADXEE	CALLED
3	PRINT-NEW-UNIT	
4	LLINK	CALLED
4	PRUNE	CALLED
4	WRACE	CALLED
3	ADD-CLEANUP-PROCESSING	
4	OBSDE	CALLED
4	PRMSE	CALLED
3	RLFLE	CALLED

SEGMENT NAME: ADXEE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	ADXE	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
2	DATA DIVISION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	OBTAIN-FIRST-BLOCK-NBR	
4	RDBKE	CALLED
4	PRMSE	CALLED
3	RDBKE	CALLED
3	INSERT-ENTRY-IN-BLOCK	
4	ASBKE	CALLED
4	SPLIT-INDEX-BLOCK	
5	WRBKE	CALLED
4	WRBKE	CALLED
3	PRMSE	CALLED

SEGMENT NAME: BCTLE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	BCTL	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
3	INPUT-OUTPUT SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	LLINK	CALLED
3	OBUSE	CALLED
3	OBFNE	CALLED
3	LLINK	CALLED
3	PROCESS-FUNCTION	
4	ADUNE	CALLED
4	PRAUE	CALLED
4	BKLBE	CALLED
4	CHUNE	CALLED
4	CMMLE	CALLED
4	CRSCE	CALLED
4	SCSGE	CALLED
4	PRDCE	CALLED
4	EXPGE	CALLED
4	PRXXE	CALLED
4	ITPJE	CALLED
4	WRJBE	CALLED
4	LKPGE	CALLED
4	ITCLE	CALLED
4	FMMDE	CALLED
4	PRMRE	CALLED
4	UPMDE	CALLED
4	MVUNE	CALLED
4	ESPAE	CALLED
4	PGUNE	CALLED
4	RPUNE	CALLED

SEGMENT NAME: BCTLE

UNIT LEVEL	UNIT NAME	UNIT TYPE
4	RSLBE	CALLED
4	PRSDE	CALLED
4	PGSCE	CALLED
4	OBKWE	CALLED
4	PRMSE	CALLED
4	OBSDE	CALLED
3	OBFNE	CALLED
3	PRMSE	CALLED
3	SPAWN-A-JOB	
4	WRITE-USERID-CARD	
5	GET-USERID-PASSWORD	
6	ASFLE	CALLED
6	LLINK	CALLED
6	RDXXE	CALLED
4	LLINK	CALLED
4	SPJBE	CALLED
4	PRMSE	CALLED
4	PRMSE	CALLED
3	PRMSE	CALLED
3	RLAFE	CALLED

SEGMENT NAME: BKLBE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	BKLB	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	VALIDATE-KEYWORDS	
4	OBKWE	CALLED
4	PRINT-MESSAGE	
5	PRMSE	CALLED
4	INTERPRET-KEYWORD-VALUES	
5	PRINT-MESSAGE	
6	PRMSE	CALLED
4	PRINT-MESSAGE	
5	PRMSE	CALLED
4	EDIT-INPUT-VALUES	
5	PRINT-MESSAGE	
6	PRMSE	CALLED
5	PRINT-MESSAGE	
6	PRMSE	CALLED
5	PRINT-MESSAGE	
6	PRMSE	CALLED
5	PRINT-MESSAGE	
6	PRMSE	CALLED
4	BACKUP-LIBRARY	
5	ASFLE	CALLED
5	INDEX-ENTRY-PROCESSING	
6	LLINK	CALLED
6	RDXXE	CALLED
6	BACKUP-SECTION	
7	ASFLE	CALLED
7	LLINK	CALLED
7	BKSCE	CALLED
7	RLFLE	CALLED

SEGMENT NAME: BKLBE

UNIT LEVEL	UNIT NAME	UNIT TYPE
5	RLFLE	CALLED
4	BACKUP-SECTION	
5	ASFLE	CALLED
5	LLINK	CALLED
5	BKSCE	CALLED
5	RLFLE	CALLED
4	PRINT-MESSAGE	
5	PRMSE	CALLED

SEGMENT NAME: BKSCE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	BKSCE	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
3	INPUT-OUTPUT SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	WRITE-SECTION-HEADER	
4	RDBKE	CALLED
3	UNIT-PROCESSING	
4	LLINK	CALLED
4	RDXXE	CALLED
4	RDBKE	CALLED
4	WRITE-FMS-PARAMETERS	
5	LLINK	CALLED
5	QYFIE	CALLED
5	BIT-PROCESSING	
6	CHECK-IF-BITS-ARE-ON	
6	CHECK-IF-BITS-ARE-ON	
6	CHECK-IF-BITS-ARE-ON	
6	CHECK-IF-BITS-ARE-ON	
6	CHECK-IF-BITS-ARE-ON	
6	CHECK-IF-BITS-ARE-ON	
5	PRMSE	CALLED
4	WRITE-INDEP-LOAD-UNIT	
5	ALFLE	CALLED
5	DAFLE	CALLED
4	WRITE-INDEP-UNIT	
5	ALFLE	CALLED
5	DAFLE	CALLED

SEGMENT NAME: BKSCE

UNIT LEVEL	UNIT NAME	UNIT TYPE
4	WRITE-PSL-UNIT	
5	ITRDE	CALLED
5	RDLNE	CALLED
5	TMRDE	CALLED
4	PRMSE	CALLED
3	PRMSE	CALLED

SEGMENT NAME: CHLNE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	CHLN	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	INITIALIZE-UNIT-CHANGES	
4	LLINK	CALLED
4	STMFE	CALLED
4	CHFLE	CALLED
4	ITRDE	CALLED
4	PRMSE	CALLED
4	TMRDE	CALLED
4	INITIALIZE-ACCT-UPDATE	
5	TMRDE	CALLED
5	PRMSE	CALLED
5	PRINT-NEW-UNIT	
6	LLINK	CALLED
6	PRUNE	CALLED
6	WRACE	CALLED
3	APPLY-UNIT-CHANGES	
4	ITWRE	CALLED
4	FILL-CHANGE-TABLE	
4	APPLY-LINE-CHANGES	
5	INSERT-DATA	
6	WRITE-CURRENT-LINE	
7	WRLNE	CALLED
6	WRITE-LINE	
7	WRLNE	CALLED
7	SCAN-FOR-INCLUDE	
8	PCLUE	CALLED
8	PRMSE	CALLED
6	WRITE-LINE	
7	WRLNE	CALLED

SEGMENT NAME: CHLNE

UNIT LEVEL	UNIT NAME	UNIT TYPE
5	DELETE-OR-MODIFY-DATA	
6	SCAN-FOR-INCLUDE	
7	PCLUE	
7	PRMSE	CALLED
6	SCAN-FOR-INCLUDE	CALLED
7	PCLUE	
7	PRMSE	CALLED
6	SHIFT-DATA	CALLED
6	MODIFY-STRING-DATA	
7	SCAN-FOR-OLD-STRING	
5	SCAN-FOR-INCLUDE	
6	PCLUE	
6	PRMSE	CALLED
5	WRITE-CURRENT-LINE	CALLED
6	WRLNE	
5	WRITE-CURRENT-LINE	CALLED
6	WRLNE	
4	COMPRESS-CHANGE-TABLE	CALLED
5	PRMSE	
5	PRMSE	CALLED
5	RDLNE	CALLED
4	PRMSE	CALLED
4	TMWRE	CALLED
4	TMRDE	CALLED
3	REWRITE-UNIT	CALLED
4	ITWRE	
4	WRLNE	CALLED
4	TMWRE	CALLED
3	CHANGE-UNIT-INDEXES	CALLED
4	CHXEE	
4	RLBKE	CALLED
3	UPDATE-ACCOUNT-INFO	CALLED
4	WRACE	
		CALLED

SEGMENT NAME: CHLNE

UNIT LEVEL	UNIT NAME	UNIT TYPE
3	PRINT-NEW-UNIT	
4	LLINK	CALLED
4	PRUNE	CALLED
3	WRACE	CALLED

SEGMENT NAME: CHRCE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	CHRC	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	LLINK	CALLED
3	STORE-KEYWORD-PARAMETERS	
4	OBFNE	CALLED
4	OBKWE	CALLED
4	PRMSE	CALLED
4	INTERPRET-SUBFUNC-VALUES	
5	NUMERIC-EDIT	
5	PRMSE	CALLED
5	NUMERIC-EDIT	
5	NUMERIC-EDIT	
5	PRMSE	CALLED
5	NUMERIC-EDIT	
5	PRMSE	CALLED
5	NUMERIC-EDIT	
5	PRMSE	CALLED
5	PRMSE	CALLED
5	PRMSE	CALLED
5	PRMSE	CALLED
5	PRMSE	CALLED
5	PRMSE	CALLED
5	PRMSE	CALLED
5	PRMSE	CALLED
5	PRMSE	CALLED
4	INTERPRET-CHANGE-VALUES	
5	OBKWE	CALLED
5	NUMERIC-EDIT	
5	PRMSE	CALLED

SEGMENT NAME: CHRCE

UNIT LEVEL	UNIT NAME	UNIT TYPE
5	GET-CHANGE-SUBFUNCTION	
6	EDIT-INPUT-VALUES	
7	PRMSE	CALLED
7	PRMSE	CALLED
7	PRMSE	CALLED
7	PRMSE	CALLED
7	PRMSE	CALLED
6	EDIT-SUBFUNCTION-VALUES	
7	EDIT-COPY-VALUE7	
8	PRMSE	CALLED
8	PRMSE	CALLED
7	PRMSE	CALLED
7	PRMSE	CALLED
7	PRMSE	CALLED
7	EDIT-MODIFY-VALUES	
8	PRMSE	CALLED
8	PRMSE	CALLED
8	PRMSE	CALLED
8	PRMSE	CALLED
6	BUILD-SORT-FILE	
7	OBSDE	CALLED
7	BUILD-MODIFY-RECORDS	
8	OBSDE	CALLED
7	LOCATE-COPY-UNIT	
8	ASFLE	CALLED
8	FDXEE	CALLED
8	PRMSE	CALLED
8	PRMSE	CALLED
7	READ-COPY-UNIT	
8	ITRDE	CALLED
8	PRMSE	CALLED
8	PRMSE	CALLED
8	RDLNE	CALLED
8	RDLNE	CALLED
8	RDLNE	CALLED
8	TMRDE	CALLED
7	RLFLE	CALLED

SEGMENT NAME: CHUNE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	CHUN	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	LLINK	CALLED
3	CHRCE	CALLED
3	LOCATE-UNIT	
4	ASFLE	CALLED
4	FDXEE	CALLED
4	PRMSE	CALLED
4	PRMSE	CALLED
3	LLINK	CALLED
3	LLINK	CALLED
3	CHLNE	CALLED
3	RLFLE	CALLED
3	PRMSE	CALLED

SEGMENT NAME: CHXEE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	CHXE	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
2	DATA DIVISION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	OBTAIN-FIRST-BLOCK-NBR	
4	RDBKE	CALLED
4	WRITE-ERROR-MSG	
5	PRMSE	CALLED
5	RDBKE	CALLED
5	LOCATE-ENTRY	
3	WRBKE	CALLED
3	WRITE-ERROR-MSG	
4	PRMSE	CALLED
4	RDBKE	CALLED
4	LOCATE-ENTRY	

SEGMENT NAME: CLMDE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	CLMD	
2	CLMD-ENVIRONMENT-DIVISION	
3	CONFIGURATION SECTION	
3	INPUT-OUTPUT SECTION	
2	DATA DIVISION	
3	CLMD-FILE-SECTION	
3	WORKING-STORAGE SECTION	
2	CLMD-PROCEDURE DIVISION	
3	LLINK	CALLED
3	PPCLE	CALLED
3	LLINK	CALLED
3	PCCLE	CALLED
3	LLINK	CALLED
3	UPCLE	CALLED
3	PRMSE	CALLED
3	RLAFE	CALLED

SEGMENT NAME: CMMLE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	CMML	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
3	INPUT-OUTPUT SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	STORE-INPUT-KEYWORDS	
4	LLINK	CALLED
4	OBKWE	CALLED
4	PRMSE	CALLED
4	INTERPRET-KEYWORD-VALUES	
5	OBKWE	CALLED
5	INTERPRET-KEYWORD-LIBRARY	
5	INTERPRET-KEYWORD-PROJECT	
5	PRMSE	CALLED
3	EDIT-INPUT-VALUES	
4	PRMSE	CALLED
4	PRMSE	CALLED
4	PRMSE	CALLED
4	EDIT-PROJECT-LIB-TABLE	
3	FIND-TOP-UNIT	
4	ASFLE	CALLED
4	FDXEE	CALLED
4	RLFLE	CALLED
4	PRMSE	CALLED
3	GET-LANG-AND-COMPILE-CNT	
4	ITRDE	CALLED
4	TMRDE	CALLED
4	WRACE	CALLED
3	PREPARE-OBJECT-FILE	
4	ASFLE	CALLED
4	FDXEE	CALLED

SEGMENT NAME: CMMLE

UNIT LEVEL	UNIT NAME	UNIT TYPE
4	MODIFY-OBJECT-UNIT	
5	LLINK	CALLED
5	STMFE	CALLED
5	CHFLE	CALLED
5	ITRDE	CALLED
5	TMRDE	CALLED
5	WRACE	CALLED
4	CREATE-OBJECT-UNIT	
5	ITWRE	CALLED
5	TMWRE	CALLED
5	ADXEE	CALLED
4	RLFLE	CALLED
3	PRMSE	CALLED
3	RLFLE	CALLED
3	WRITE-JCL-FOR-COMPILE	
4	ASFLE	CALLED
4	FDXEE	CALLED
4	ITRDE	CALLED
4	PRMSE	CALLED
4	RDLNE	CALLED
4	WRITE-JOB-CARD	
5	WRITE-OBJECT-CARD	
6	FILL-FILE-STRING	
6	FILL-FILE-STRING	
6	FILL-FILE-STRING	
5	WRITE-INPUT-KEYWORD-CARDS	
5	WRITE-SOURCE-CARD	
6	FILL-FILE-STRING	
6	FILL-FILE-STRING	
6	FILL-FILE-STRING	
6	PRMSE	CALLED
4	RDLNE	CALLED
4	RLFLE	CALLED
3	PRMSE	CALLED

SEGMENT NAME: CRSCE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	CRSC	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
3	INPUT-OUTPUT SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	LLINK	CALLED
3	STORE-INPUT-KEYWORDS	
4	OBKWE	CALLED
4	PRMSE	CALLED
4	INTERPRET-KEYWORD VALUES	
5	OBKWE	CALLED
4	PRMSE	CALLED
4	EDIT-AND-SAVE-OPTIONS	
5	EDIT-SECTION-OPTIONS	
5	PRMSE	CALLED
5	PRMSE	CALLED
3	ASFLE	CALLED
3	FDXEE	CALLED
3	CREATE-NEW-SECTION	
4	STCFE	CALLED
4	CRFLE	CALLED
4	LLINK	CALLED
4	ITSFE	CALLED
4	CREATE-SECTION-CATALOG	
5	LLINK	CALLED
5	CRCAE	CALLED
4	INITIALIZE-MGMT-SECTION	
5	ASFLE	CALLED
5	WRACE	CALLED
5	ADXEE	CALLED

SEGMENT NAME: CRSCE

UNIT LEVEL	UNIT NAME	UNIT TYPE
5	RLFLE	CALLED
4	ADXEE	CALLED
4	DELETE-NEW-SECTION	CALLED
5	LLINK	CALLED
5	PGFLE	CALLED
3	CHANGE-SECTION-PARAMETERS	CALLED
4	PRMSE	CALLED
3	RLFLE	CALLED
3	PRMSE	CALLED

SEGMENT NAME: DLUNE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	DLUN	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
2	DATA DIVISION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	DELETE-UNIT	
4	RDBKE	CALLED
4	ERROR-PROCESSING-UNIT	
5	PRMSE	CALLED
4	LLINK	CALLED
4	PGFLE	CALLED
4	INCLUDE-PROCESSING	
5	ITRDE	CALLED
5	LLINK	CALLED
5	RDLNE	CALLED
5	SCAN-FOR-INCLUDE	
6	ERROR-PROCESSING	
7	PRMSE	CALLED
5	PCLUE	
4	REPLACE-UNIT-WITH-STUB	
5	WRBKE	CALLED
5	ERROR-PROCESSING	CALLED
6	PRMSE	
4	RLBKE	CALLED
4	DLXEE	CALLED
4	CHXEE	CALLED

SEGMENT NAME: DLXEE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	DXLE	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
2	DATA DIVISION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	RDBKE	
3	WRITE-ERROR-MSG	CALLED
4	PRMSE	
3	LOCATE-AND-DELETE	CALLED
4	RDBKE	
4	WRBKE	CALLED
3	WRITE-ERROR-MSG	CALLED
4	PRMSE	
		CALLED

SEGMENT NAME: ESPAE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	ESPA	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
2	DATA DIVISION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	STORE-INPUT-KEYWORDS	
4	OBKWE	CALLED
4	PRMSE	CALLED
4	INTERPRET-KEYWORD-VALUES	
4	PRMSE	CALLED
3	PRMSE	CALLED
3	OBFNE	CALLED
3	OBKWE	CALLED
3	OBSDE	CALLED
3	OBFNE	

SEGMENT NAME: EXPGE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	EXPG	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
3	INPUT-OUTPUT SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	STORE-INPUT-KEYWORDS	
4	OBKWE	CALLED
4	PRMSE	CALLED
4	INTERPRET-KEYWORD-VALUES	
5	INTERPRET-KEYWORD-LIBRARY	
5	INTERPRET-KEYWORD-PROJECT	
4	PRMSE	CALLED
3	EDIT-INPUT-VALUES	
4	PRMSE	CALLED
4	PRMSE	CALLED
4	PRMSE	CALLED
4	EDIT-PROJECT-LIB-TABLE	
4	PRMSE	CALLED
3	FIND-SECTION-FILE-UNITS	
4	FIND-TOP-UNIT	
5	ASFLE	CALLED
5	FDXEE	CALLED
5	RLFLE	CALLED
5	PRMSE	CALLED
4	RLFLE	CALLED
4	FIND-TOP-UNIT	
5	ASFLE	CALLED
5	FDXEE	CALLED
5	RLFLE	CALLED
5	PRMSE	CALLED

SEGMENT NAME: EXPGE

UNIT LEVEL	UNIT NAME	UNIT TYPE
4	WRITE-EXPG-JCL	
5	WRITE-JCL-FOR-PREP-LINK	
6	ASFLE	CALLED
6	FDXEE	CALLED
6	ITRDE	CALLED
6	PRMSE	CALLED
6	RDLNE	CALLED
6	WRITE-PPLK-CARD	
7	WRITE-INPUT-KEYWORD-CARDS	
6	RDLNE	CALLED
6	RLFLE	CALLED
4	WRITE-JCL-FOR-EXECUTE	
5	ITRDE	CALLED
5	RDLNE	CALLED
5	WRITE-JOB-CARD	
6	WRITE-LOAD-CARD	
7	FILL-FILE-STRING	
7	FILL-FILE-STRING	
7	FILL-FILE-STRING	
6	WRITE-LOAD-CARD	
7	FILL-FILE-STRING	
7	FILL-FILE-STRING	
7	FILL-FILE-STRING	
6	WRITE-EXECUTE-CARD	
7	FILL-FILE-STRING	
7	FILL-FILE-STRING	
5	RDLNE	CALLED
5	WRITE-JOB-CARD	
6	WRITE-LOAD-CARD	
7	FILL-FILE-STRING	
7	FILL-FILE-STRING	
7	FILL-FILE-STRING	
6	WRITE-LOAD-CARD	
7	FILL-FILE-STRING	
7	FILL-FILE-STRING	

SEGMENT NAME: EXPGE

UNIT LEVEL	UNIT NAME	UNIT TYPE
7	FILL-FILE--STRING	
6	WRITE-EXECUTE-CARD	
7	FILL-FILE-STRING	
7	FILL-FILE-STRING	
3	RLFLE	CALLED
3	PRMSE	CALLED

SEGMENT NAME: FDXEE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	FDXE	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
2	DATA DIVISION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	RDBKE	CALLED
3	WRITE-ERROR-MSG	
4	PRMSE	CALLED
3	LOCATE-REQUESTED-ENTRY	
4	RDBKE	CALLED
3	WRITE-ERROR-MSG	
4	PRMSE	CALLED

SEGMENT NAME: FMEDE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	FMEDE	
2	FMED-ENVIRONMENT-DIVISION	MAIN
2	FMED-FILE-SECTION	
2	FMED-WORKING-STORAGE-77	
2	FMED-WORKING-STORAGE-01	
3	CLIB-CODES-FOR-FMMD	
3	CLIB-MDCR-DATA-RECORD-FORMAT	
2	FMED-PROCEDURE-DIVISION	
3	FMED-EDIT-SORT-TRANSACTIONS	
4	OBSDE	
4	FMED-EDIT-SPEC-OP-FLD	CALLED
4	FMED-MOVE-WRITE-TRANSACTION	
4	PRMSE	CALLED

SEGMENT NAME: FMMDE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	FMMD	
2	FMMD-ENVIRONMENT-DIVISION	
3	CONFIGURATION SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	FMMD-STORE-EDIT-KEYWORDS	
4	OBKWE	CALLED
4	PRMSE	CALLED
4	FMMD-INTERPRET-FMMD-VALUE	
4	FMMD-INTERPRET-DATA-VALUE	
5	FMMD-NUMERIC-EDIT	
5	FMMD-NUMERIC-EDIT	
4	PRMSE	CALLED
4	FMMD-CHECK-FMMD-VALUES	
5	PRMSE	CALLED
5	PRMSE	CALLED
5	PRMSE	CALLED
5	PRMSE	CALLED
3	FMMD-ACCESS-MGMT-DATA-FILE	CALLED
4	ASFLE	CALLED
4	FDXEE	CALLED
4	PRMSE	CALLED
3	FMMD-CALL-MOVE-OLDPROJ-OLDLIB	CALLED
4	LINK	
4	FMMVE	CALLED
3	FMMD-READ-EDIT-SORT-TRANS	CALLED
4	LLINK	
4	FMEDE	CALLED
4	OBSDE	CALLED
4	LLINK	CALLED

SEGMENT NAME: FMMDE

UNIT LEVEL	UNIT NAME	UNIT TYPE
3	ITWRE	CALLED
3	LLINK	CALLED
3	FMUPE	CALLED
3	TMWRE	CALLED
3	FMMD-CLEANUP-WRITE-ACCT-INFO	
4	ADXEE	CALLED
4	CHXEE	CALLED
4	RLBKE	CALLED
4	FMMD-SETUP-ACCOUNTING-INFO	
4	WRACE	CALLED
4	LLINK	CALLED
4	PRUNE	CALLED
4	WRACE	CALLED
3	FMMD-DELETE-MGMT-INFO	
4	LLINK	CALLED
4	DLUNE	CALLED
3	RLFLE	CALLED
3	OBSDE	CALLED

SEGMENT NAME: FMMVE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	FMMV	
2	FMMV-ENVIRONMENT-DIVISION	
3	CONFIGURATION SECTION	
2	DATA DIVISION	
3	FMMV-FILE-SECTION	
3	WORKING-STORAGE SECTION	
2	FMMV-PROCEDURE-DIVISION	
3	FMMV-MOVE-OLDPROJ-OLDLIB	
4	ASFLE	
4	FDXEE	CALLED
4	ITRDE	CALLED
4	FMMV-MOVE-FORMAT	CALLED
5	ITWRE	
5	RDLNE	CALLED
5	WRLNE	CALLED
5	TMWRE	CALLED
4	TMRDE	CALLED
4	PRMSE	CALLED
4	RLFLE	CALLED
		CALLED

SEGMENT NAME: FMUPE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	FMUP	
2	FMUP-ENVIRONMENT-DIVISION	
3	CONFIGURATION SECTION	
2	DATA DIVISION	
3	FMUP-FILE-SECTION	
3	WORKING-STORAGE SECTION	
2	FMUP-PROCEDURE-DIVISION	
3	FMUP-INITIALIZE-ADD-MGMT-UNIT	
4	WRLNE	CALLED
3	FMUP-CHANGE-MGMT-UNIT	
4	ITRDE	CALLED
4	RDLNE	CALLED
4	WRLNE	CALLED
4	RDLNE	CALLED
4	FMUP-ADD-CHANGE-TEMP-DATA	
5	WRLNE	CALLED
5	PRMSE	CALLED
5	RDLNE	CALLED
5	WRLNE	CALLED
5	WRLNE	CALLED
5	RDLNE	CALLED
4	FMUP-ADD-LAST-INSERTS	
5	WRLNE	CALLED
5	PRMSE	CALLED
4	TMRDE	CALLED

SEGMENT NAME: ITCLE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	ITCLE	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
3	INPUT-OUTPUT SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	ITCL-PROCEDURE-DIVISION	
3	ITCL-STORE-EDIT-INPUT-KEYWORD	
4	OBKWE	CALLED
4	PRMSE	CALLED
4	ITCL-INTERPRET-KEYWORD-VALUES	
4	PRMSE	CALLED
4	PRMSE	CALLED
4	PRMSE	CALLED
3	ASFLE	CALLED
3	ITCL-CHECK-COLLECTION-UNIT-KEY	
4	FDXEE	CALLED
4	ITRDE	CALLED
4	PRMSE	CALLED
3	RLFLE	CALLED
3	ITCL-WRITE-JCL-FOR-MDCOLLECT	
4	ASFLE	CALLED
4	FDXEE	CALLED
4	ITRDE	CALLED
4	PRMSE	CALLED
4	RDLNE	CALLED
4	ITCL-WRITE-INPUT-KEYWORD-CARDS	
3	RDLNE	CALLED
3	PRMSE	CALLED

SEGMENT NAME: ITPJE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	ITPJ	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
3	INPUT-OUTPUT SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	LLINK	CALLED
3	STORE-INPUT-KEYWORDS	
4	OBKWE	CALLED
4	PRMSE	CALLED
4	INTERPRET-KEYWORD-VALUES	
5	OBKWE	CALLED
4	PRMSE	CALLED
4	PRMSE	CALLED
4	PRMSE	CALLED
3	INITIALIZE-PROJECT-INDEX	
4	STCFE	CALLED
4	CPFLE	CALLED
4	LLINK	CALLED
4	ITSFE	CALLED
3	ADD-PASSWORD-TO-DIRECTORY	
4	ASFLE	CALLED
4	ADXEE	CALLED
4	RLFLE	CALLED
3	ADD-PROJ-TO-SYSTEM-INDEX	
3	CHANGE-PROJECT-PASSWORD	
4	ASFLE	CALLED
4	LLINK	CALLED
4	RDXXE	CALLED
4	DLXEE	CALLED
4	ADXEE	CALLED
4	RLFLE	CALLED
3	LLINK	CALLED
3	PGFLE	CALLED
3	PRMSE	CALLED

SEGMENT NAME: ITRDE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	RDUN	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
3	INPUT-OUTPUT SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	INITIALIZE-TO-READ	
4	ERROR-PROCESSING	
5	PRMSE	CALLED
4	ERROR-PROCESSING	
5	PRMSE	CALLED
4	FDXEE	CALLED
4	ERROR-PROCESSING	
5	PRMSE	CALLED
4	RDBKE	CALLED
4	INITIALIZE-SEQ-UNIT	
5	ASFLE	CALLED
4	INITIALIZE-RANDOM-UNIT	
4	ERROR-PROCESSING	
5	PRMSE	CALLED

SEGMENT NAME: RDLNE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	RDUN	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
3	INPUT-OUTPUT SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	READ-A-LINE	
4	GET-ANOTHER-BLOCK	
5	RDBKE	CALLED
5	RDBKE	CALLED
4	READ-COMPRESSED-LINE	
4	ERROR-PROCESSING	
5	PRMSE	CALLED
4	RLFLE	CALLED
4	ERROR-PROCESSING	
5	PRMSE	CALLED

SEGMENT NAME: TMRDE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	RDUN	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
3	INPUT-OUTPUT SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	TERMINATE-READ	
4	RDBKE	CALLED
4	RLFLE	CALLED
4	ERROR-PROCESSING	
5	PRMSE	CALLED

SEGMENT NAME: ITSFE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	ITSF	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	CALCULATE-NBRS-OF-BLOCKS	
3	ALFLE	CALLED
3	INIT-FIRST-CONTROL-BLOCK	
4	TURN-ON-FREE-BLOCKS-SWS	
4	TURN-OFF-USED-BLOCKS-SWS	
5	LLINK	CALLED
5	ANBTE	CALLED
4	PRMSE	CALLED
3	INIT-INDEX-BLOCKS	
4	PRMSE	CALLED
3	INIT-CONT-CONTROL-BLOCKS	
4	TURN-ON-FREE-BLOCK-SWS	
4	PRMSE	CALLED
3	INIT-REMAINING-BLOCKS	
4	PRMSE	CALLED
3	DAFLE	CALLED
3	PRMSE	CALLED

SEGMENT NAME: ITWRE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	WRUN	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
3	INPUT-OUTPUT SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	INITIALIZED-TO-WRITE	
4	ASBKE	CALLED
4	RDBKE	CALLED
4	INITIALIZE-SEQ-UNIT	
5	LLINK	CALLED
5	STCFE	CALLED
5	LLINK	CALLED
5	CRFLE	CALLED
5	ASFLE	CALLED
4	INITIALIZE-RANDOM-UNIT	
5	RLBKE	CALLED
5	ERROR-PROCESSING	
6	PRMSE	CALLED

SEGMENT NAME: TMWRE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	WRUN	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
3	INPUT-OUTPUT SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	TERMINATE-WRITE	CALLED
4	RLBKE	
4	ERROR-PROCESSING	CALLED
5	PRMSE	CALLED
4	WRBKE	CALLED
4	RLFLE	
4	ERROR-PROCESSING	CALLED
5	PRMSE	

SEGMENT NAME: WRLNE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	WRUN	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
3	INPUT-OUTPUT SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	WRITE-A-LINE	
4	SCAN-FOR-DATA	
5	ASBKE	CALLED
5	ERROR-PROCESSING	
6	PRMSE	CALLED
5	WRBKE	CALLED
5	RDBKE	CALLED
5	ERROR-PROCESSING	
6	PRMSE	CALLED
4	WRITE-COMPRESSED-LINE	
4	GET-ANOTHER-BLOCK	
5	ASBKE	CALLED
5	ERROR-PROCESSING	
6	PRMSE	CALLED
5	WRBKE	CALLED
5	RDBKE	CALLED
5	ERROR-PROCESSING	
6	PRMSE	CALLED

SEGMENT NAME: LKPGE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	LKPG	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
3	INPUT-OUTPUT SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	LINK-PROGRAM	
4	STORE-INPUT-KEYWORDS	CALLED
5	LLINK	CALLED
5	OBKWE	CALLED
5	PRMSE	
5	INTERPRET-KEYWORD-VALUES	CALLED
6	OBKWE	
6	INTERPRET-KEYWORD-LIBRARY	
6	INTERPRET-KEYWORD-PROJECT	CALLED
5	PRMSE	
4	EDIT-INPUT-VALUES	CALLED
5	PRMSE	CALLED
5	PRMSE	CALLED
5	PRMSE	
5	EDIT-PROJECT-LIB-TABLE	
4	FIND-TOP-UNIT	CALLED
5	ASFLE	CALLED
5	FDXEE	CALLED
5	RLFLE	CALLED
5	PRMSE	CALLED
4	RLFLE	
4	PREPARE-LOAD-FILE	CALLED
5	ASFLE	CALLED
5	FDXEE	
5	MODIFY-LOAD-UNIT	

SEGMENT NAME: LKPGE

UNIT LEVEL	UNIT NAME	UNIT TYPE
6	LLINK	CALLED
6	STMFE	CALLED
6	CHFLE	CALLED
6	RDBKE	CALLED
6	WRACE	CALLED
5	CREATE-LOAD-UNIT	
6	STCFE	CALLED
6	CRFLE	CALLED
6	WRACE	CALLED
6	ADXEE	CALLED
6	RLBKE	CALLED
5	RLFLE	CALLED
4	WRITE-JCL-FOR-LINE	
5	ASFLE	CALLED
5	FDXEE	CALLED
5	ITRDE	CALLED
5	PRMSE	CALLED
5	RDLNE	CALLED
5	WRITE-JOB-CARD	
6	WRITE-LOAD-CARD	
7	FILL-FILE-STRING	
7	FILL-FILE-STRING	
7	FILL-FILE-STRING	
6	WRITE-INPUT-KEYWORD-CARDS	
5	RDLNE	CALLED
4	RLFLE	
4	PRMSE	

SEGMENT NAME: MVUNE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	MVUN	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
3	INPUT-OUTPUT SECTION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	LLINK	CALLED
3	STORE-KEYWORDS	
4	OBKWE	CALLED
4	PRMSE	CALLED
4	INTERPRET-KEYWORDS	CALLED
5	OBKWE	CALLED
4	PRMSE	CALLED
4	EDIT-MOVE-KEYWORDS	
5	PRMSE	CALLED
5	PRMSE	CALLED
5	PRMSE	CALLED
5	PRMSE	CALLED
5	PRMSE	CALLED
5	PRMSE	CALLED
5	PRMSE	CALLED
3	ASSIGN-SECTIONS	
4	ASFLE	CALLED
4	ASFLE	CALLED
3	LLINK	CALLED
3	FIND-LIBRARY-UNITS	
4	RDXEE	CALLED
4	FIND-RECEIVING-UNIT	
5	FDXEE	CALLED
4	PRMSE	CALLED
4	PRMSE	CALLED
4	PRMSE	CALLED
3	INITIALIZE-UNIT-MOVE	

SEGMENT NAME: MVUNE

UNIT LEVEL	UNIT NAME	UNIT TYPE
5	WRACE	CALLED
5	LLINK	CALLED
5	PRUNE	CALLED
5	PRMSE	CALLED
5	RLBKE	CALLED
3	FIND-LIBRARY-UNITS	
4	RDXXE	CALLED
4	FIND-RECEIVING-UNIT	
5	FDXEE	CALLED
5	PRMSE	CALLED
4	PRMSE	CALLED
4	PRMSE	CALLED
3	FIND-SINGLE-UNIT	
4	FDXEE	CALLED
4	FIND-RECEIVING-UNIT	
5	FDXEE	CALLED
5	PRMSE	CALLED
4	PRMSE	CALLED
4	PRMSE	CALLED
3	INITIALIZE-UNIT-MOVE	
4	OBTAIN-SENDING-ACCT-INFO	
5	ITRDE	CALLED
5	COMPARE-KEYS-AND-UTYPES	
6	PRMSE	CALLED
6	PRMSE	CALLED
6	PRMSE	CALLED
5	TMRDE	CALLED
5	ITRDE	CALLED
4	SETUP-ACCOUNTING-INFO	
4	LLINK	CALLED
4	STMFE	CALLED
4	CHFLE	CALLED
4	ITWRE	CALLED
4	TMRDE	CALLED

SEGMENT NAME: MVUNE

UNIT LEVEL	UNIT NAME	UNIT TYPE
4	PRMSE	CALLED
4	TMWRE	CALLED
4	PRMSE	CALLED
3	UNIT-MOVE	
4	LLINK	CALLED
4	RDLNE	CALLED
4	STORE-LOWER-UNIT-INFO	
5	PCLUE	CALLED
5	PRMSE	CALLED
4	WRLNE	CALLED
4	PCLUE	CALLED
4	RDLNE	CALLED
4	TMWRE	CALLED
4	TMRDE	CALLED
4	UPDATE-ACCOUNT-INFO	
4	CONCLUDE-UNIT-MOVE	
5	LLINK	CALLED
5	DLUNE	CALLED
5	CHXEE	CALLED
5	ADXEE	CALLED
5	WRACE	CALLED
5	LLINK	CALLED
5	PRUNE	CALLED
5	PRMSE	CALLED
5	RLBKE	CALLED
3	TERMINATE-MOVE-PROCESSING	
4	RLFLE	CALLED
4	PRMSE	CALLED

SEGMENT NAME: OBFNE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	OBIC	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
3	INPUT-OUTPUT SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	INPUT-CARD-FUNCTION	
4	CARD-FORMAT-CHECK	
5	PRMSE	CALLED
5	PRINT-CARD	
6	PRMSE	CALLED
5	PRMSE	CALLED
4	OBTAIN-FUNCTION-WORD	
5	PRINT-CARD	
6	PRMSE	CALLED
5	PRMSE	CALLED
5	PRINT-CARD	
6	PRMSE	CALLED
5	PRMSE	CALLED
4	INIT-FUNC-KW-PROCESS	
4	PRINT-CARD	
5	PRMSE	CALLED
4	READ-INPUT-CARD	
4	READ-INPUT-CARD	

SEGMENT NAME: OBKWE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	OBIC	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
3	INPUT-OUTPUT SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	GET-KEYWORD	
4	PRINT-CARD	
5	PRMSE	CALLED
4	READ-INPUT-CARD	
4	PRMSE	CALLED
4	INTERPRET-FIRST-WORD	
5	GET-WORD-DELIMITER-PAIR	
6	GET-VALUE-BETWEEN-QUOTES	
6	PRMSE	CALLED
5	STORE-1ST-WORD-DELIMITER	
6	PRMSE	CALLED
5	SCAN-TO-NEXTWORD	
6	SCAN-DELIMITER	
7	READ-INPUT-CARD	
7	CARD-FORMAT-CHECK	
8	PRMSE	CALLED
8	PRINT-CARD	
9	PRMSE	
8	PRMSE	CALLED
7	PRMSE	CALLED
6	READ-INPUT-CARD	
6	CARD-FORMAT-CHECK	
7	PRMSE	CALLED
7	PRINT-CARD	
8	PRMSE	CALLED
7	PRMSE	CALLED

SEGMENT NAME: OBKWE

UNIT LEVEL	UNIT NAME	UNIT TYPE
6	PRMSE	
6	READ-INPUT-CARD	CALLED
4	INTERPRET-SECOND-WORD	
5	GET-WORD-DELIMITER-PAIR	
6	GET-VALUE-BETWEEN-QUOTES	
6	PRMSE	
5	STORE-2ND-WORD-DELIMITER	CALLED
6	PRMSE	
5	SCAN-TO-NEXTWORD	CALLED
6	SCAN-DELIMITER	
7	READ-INPUT-CARD	
7	CARD-FORMAT-CHECK	
8	PRMSE	
8	PRINT-CARD	CALLED
9	PRMSE	
8	PRMSE	CALLED
7	PRMSE	CALLED
6	READ-INPUT-CARD	CALLED
6	CARD-FORMAT-CHECK	
7	PRMSE	
7	PRINT-CARD	CALLED
8	PRMSE	
7	PRMSE	CALLED
6	PRMSE	CALLED
6	READ-INPUT-CARD	CALLED
4	INTERPRET-THIRD-WORD	
5	GET-WORD-DELIMITER-PAIR	
6	GET-VALUE-BETWEEN-QUOTES	
6	PRMSE	
5	STORE-3RD-WORD-DELIMITER	CALLED
6	PRMSE	
5	SCAN-TO-NEXTWORD	CALLED
6	SCAN-DELIMITER	

SEGMENT NAME: OBKWE

UNIT LEVEL	UNIT NAME	UNIT TYPE
7	READ-INPUT-CARD	
7	CARD-FORMAT-CHECK	CALLED
8	PRMSE	
8	PRINT-CARD	CALLED
9	PRMSE	CALLED
8	PRMSE	CALLED
7	PRMSE	
6	READ-INPUT-CARD	
6	CARD-FORMAT-CHECK	CALLED
7	PRMSE	
7	PRINT-CARD	CALLED
8	PRMSE	CALLED
7	PRMSE	CALLED
6	PRMSE	
6	READ-INPUT-CARD	

SEGMENT NAME: OBSDE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	OBIC	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
3	INPUT-OUTPUT SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	INPUT-CARD-FUNCTION	
4	CARD-FORMAT--CHECK	
5	PRMSE	CALLED
5	PRINT-CARD	
6	PRMSE	CALLED
5	PRMSE	CALLED
4	OBTAIN-FUNCTION-WORD	
5	PRINT-CARD	
6	PRMSE	CALLED
5	PRMSE	CALLED
5	PRINT-CARD	
6	PRMSE	CALLED
5	PRMSE	CALLED
4	INIT-FUNC-KW-PROCESS	
4	PRINT-CARD	
5	PRMSE	CALLED
4	READ-INPUT-CARD	
4	READ-INPUT-CARD	
3	PRINT-CARD	
4	PRMSE	CALLED
3	READ-INPUT-CARD	
3	FILL-SOURCE DATA	
4	PRMSE	CALLED
4	READ-INPUT-CARD	
3	CARD-FORMAT-CHECK	

SEGMENT NAME: OBSDE

UNIT LEVEL	UNIT NAME	UNIT TYPE
4	PRMSE	CALLED
4	PRINT-CARD	CALLED
5	PRMSE	CALLED
4	PRMSE	
3	SCAN-FOR-ENDATA	
4	INIT-FUNC-KW-PROCESS	
4	PRINT-CARD	CALLED
5	PRMSE	
4	READ-INPUT-CARD	
3	FILL-SOURCE-DATA	CALLED
4	PRMSE	
4	READ-INPUT-CARD	

SEGMENT NAME: PCCLE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	PCCL	
2	CLMD-ENVIRONMENT-DIVISION	
3	CONFIGURATION SECTION	
3	INPUT-OUTPUT SECTION	
2	DATA DIVISION	
3	CLMD-FILE-SECTION	
3	WORKING-STORAGE SECTION	
2	PCCL-PROCEDURE-DIVISION	
3	CLMD-ASSIGN-MGMT-SECTION-FILE	
4	ASFLE	CALLED
3	CLMD-BUILD-FORMATS-TABLE	
4	INITIALIZE-READ-MGMT-UNIT	
5	FDXEE	
5	ITRDE	
5	PRMSE	
4	CLMD-DETERMINE-CYCLE-REQMT	
5	CLMD-DETERMINE-IF-END-OF-CYCLE	
5	WRACE	
4	CLMD-READ-FORMAT-UNIT	
5	RDLNE	CALLED
5	PRMSE	CALLED
5	RDLNE	CALLED
5	TMRDE	CALLED
4	CLMD-VERIFY-FORMAT-UNIT	
5	PRMSE	CALLED
4	CLMD-CHECK-IF-TABLE-EXCEEDED	
5	PRMSE	
4	CLMD-COLLECT-MGMTS-STATISTICS	
5	CLMD-PROCESS-PLAN-UNITS	
6	CLMD-INITIALIZE-OUTPUT-CONTROL	
7	CLMD-GET-RANDOM-STORED-DATA	
8	PRMSE	CALLED
5	CLMD-COLLECT-MODULE-STATISTICS	

NAME: PCCLE

UNIT LEVEL	UNIT NAME	UNIT TYPE
6	RLFLE	CALLED
6	CLMD-ASSIGN-SOURCE-LIBRARIES	
7	ASFLE	CALLED
7	ASFLE	CALLED
6	CLMD-FIND-SOURCE-UNIT	
7	FDXEE	CALLED
7	PRMSE	CALLED
6	ITRDE	CALLED
6	CLMD-PROCESS-UNIT-ACCOUNT-INFO	
7	TMRDE	CALLED
7	CLMD-COLLECT-UNIT-STATISTICS	
8	CLMD-PERFORM-SUMMARY-FUNCTION	
7	CLMD-OUTPUT-RECYCLE-UNIT-INFO	
8	CLMD-GET-UNIT-ITEM-VALUE	
9	CLMD-CONVERT-UNIT-TYPE-VALUE	
8	CLMD-SEARCH-SPECIAL-INPUTS	
8	WRACE	CALLED
6	RDLNE	CALLED
6	CLMD-SCAN-FOR-INCLUDE-UNIT	
6	TMRDE	CALLED
6	CLMD-CHECK-IF-TABLE-EXCEEDED	
7	PRMSE	CALLED
6	CLMD-RELEASE-SOURCE-LIBRARIES	
7	RLFLE	CALLED
5	CLMD-COLLECT-MGMT-INPUTS	
6	CLMD-ASSIGN-MGMT-SECTION-FILE	
7	ASFLE	CALLED
6	INITIALIZE READ-MGMT-UNIT	
7	FDXEE	CALLED
7	ITRDE	CALLED
7	PRMSE	CALLED
6	RDLNE	CALLED
6	RDLNE	CALLED

SEGMENT NAME: PCCLE

UNIT LEVEL	UNIT NAME	UNIT TYPE
6	TMRDE	CALLED
5	CLMD-COLLECT-CYCLE-STATISTICS	
6	GET-RANDOM-ITEM-VALUE	
7	PRMSE	CALLED
6	GET-RANDOM-ITEM-VALUE	
7	PRMSE	CALLED
5	CLMD-OUTPUT-COLLECTED-DATA	
6	CLMD-ADJUST-FORMAT-TBL-VALUES	
6	CLMD-POSITION-MGMT-FILE-INPUTS	
7	ITRDE	CALLED
7	RDLNE	CALLED
6	CLMD-SEARCH-ACCOUNTING-INPUTS	
7	CLMD-CONVERT-UNIT-TYPE-VALUE	
6	CLMD-SEARCH-SPECIAL-INPUTS	
6	CLMD-RANDOM-STORE-ITEM-VALUE	
7	PRMSE	CALLED
7	PRMSE	CALLED
6	CLMD-SEARCH-ACCOUNTING-INPUTS	
7	CLMD-CONVERT-UNIT-TYPE-VALUE	
6	CLMD-SEARCH-SPECIAL-INPUTS	
6	CLMD-SEARCH-MGMT-INPUT-DATA	
7	CLMD-PRINT-XCHECK-ERROR	
8	PRMSE	CALLED
7	CLMD-PERFORM-EXCEPTION-CHECK	
8	CLMD-PRINT-XCHECK-ERROR	
9	PRMSE	CALLED
7	CLMD-PRINT-XCHECK-ERROR	
8	PRMSE	CALLED
7	RDLNE	CALLED
7	TMRDE	CALLED
7	CLMD-CLOSE-OUT-COLLECTED-DATA	
8	PRMSE	CALLED
5	CLMD-ACCUMULATE-SUMMARY-DATA	

SEGMENT NAME: PCCLE

UNIT LEVEL	UNIT NAME	UNIT TYPE
6	CLMD-PERFORM-SUMMARY-FUNCTION	
5	CLMD-INITIALIZE-OUTPUT-CONTROL	
6	CLMD-GET-RANDOM-STORED-DATA	
7	PRMSE	
3	RLFLE	CALLED
3	CLMD-INITIALIZE-READ-MGMT-UNIT	
4	FDXEE	CALLED
4	ITRDE	CALLED
4	PRMSE	CALLED
3	CLMD-GET-RANDOM-ITEM-VALUE	
4	PRMSE	CALLED

SEGMENT NAME: PCLUE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	PCLU	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
2	DATA DIVISION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	FDXEE	CALLED
3	UPDATE-EXISTING-LU	
4	ITRDE	CALLED
4	TMRDE	CALLED
4	ADD-TO-LU-ACCTG-INFO	
5	BUILD-INITIAL-HUB	
6	ASBKE	CALLED
6	WRBKE	CALLED
4	DEL-FROM-SINGLY-INCL-UNIT	
5	PRMSE	CALLED
5	DLXEE	CALLED
5	RLBKE	CALLED
4	CHG-TO-SINGLY-INCL-UNIT	
5	RDBKE	CALLED
5	RLBKE	CALLED
4	UPDATE-HU-LIST	
5	ASBKE	CALLED
5	RDBKE	CALLED
5	PROCESS-NEXT-HU	
6	ASBKE	CALLED
6	WRBKE	CALLED
5	FINISH-HU-PROCESS	
6	WRBKE	CALLED
6	RLBKE	CALLED
4	WRACE	CALLED
4	CHXEE	CALLED

SEGMENT NAME: PCLUE

UNIT LEVEL	UNIT NAME	UNIT TYPE
3	ADD-STUB-UNIT	
4	INITIALIZE-FOR-STUB-UNIT	
4	WRACE	CALLED
4	ADXEE	CALLED
4	RLBKE	CALLED
3	PRMSE	CALLED
3	PRMSE	CALLED

SEGMENT NAME: PGSCE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	PGSC	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
3	INPUT-OUTPUT SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	VALIDATE-KEYWORDS	
4	OBKWE	CALLED
4	PRINT-MESSAGE	
5	PRMSE	CALLED
4	INTERPRET-KEYWORD-VALUES	
5	PRINT-MESSAGE	
6	PRMSE	CALLED
4	PRINT-MESSAGE	
5	PRMSE	CALLED
4	EDIT-INPUT-VALUES	
5	PRINT-MESSAGE	
6	PRMSE	CALLED
5	PRINT-MESSAGE	
6	PRMSE	CALLED
5	PRINT-MESSAGE	
6	PRMSE	CALLED
5	PRINT-MESSAGE	
6	PRMSE	CALLED
3	ASSIGN-PROJECT-FILE	
4	ASFLE	CALLED
3	INDEX-ENTRY-PROCESSING	
4	LLINK	CALLED
4	RDXXE	CALLED
4	TERMINATE-SECTION	

SEGMENT NAME: PGSCE

UNIT LEVEL	UNIT NAME	UNIT TYPE
5	ASFLE	CALLED
5	LLINK	CALLED
5	BKSCE	CALLED
5	LLINK	CALLED
5	RDXXE	CALLED
5	LLINK	CALLED
5	PGFLE	CALLED
5	LLINK	CALLED
5	PGFLE	CALLED
5	PLXEE	CALLED
5	PRINT-MESSAGE	
6	PRMSE	CALLED
5	RLFLE	CALLED
3	TERMINATE-SECTION	
4	ASFLE	CALLED
4	LLINK	CALLED
4	BKSCE	CALLED
4	LLINK	CALLED
4	RDXXE	CALLED
4	LLINK	CALLED
4	PGFLE	CALLED
4	LLINK	CALLED
4	PGFLE	CALLED
4	DLXEE	CALLED
4	PRINT-MESSAGE	CALLED
5	PRMSE	
4	RLFLE	CALLED
3	RLAFE	CALLED
3	LLINK	CALLED
3	PGFLE	CALLED
3	PRINT-MESSAGE	
4	PRMSE	CALLED

SEGMENT NAME: PGUNE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	PGUN	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
2	DATA DIVISION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	VALIDATE-INPUT-KEYWORDS	
4	OBKWE	CALLED
4	ERROR-PROCESSING	
5	PRMSE	CALLED
4	INTERPRET-KEYWORD-VALUES	
5	ERROR-PROCESSING	
6	PRMSE	
4	EDIT-INPUT-VALUES	
5	ERROR-PROCESSING	
6	PRMSE	CALLED
5	ERROR-PROCESSING	
6	PRMSE	CALLED
5	ERROR-PROCESSING	
6	PRMSE	CALLED
5	ERROR-PROCESSING	
6	PRMSE	CALLED
4	MOVE-PARAM-TABLE-ENTRIES	
5	ASFLE	CALLED
3	FIND-AND-DELETE-UNIT	
4	FDXEE	CALLED
4	LLINK	CALLED
4	DLUNE	CALLED
4	DLUNE	CALLED
4	ERROR-PROCESSING	
5	PRMSE	CALLED
4	PURGE-INDEPENDENT-FILE	
5	LLINK	CALLED
5	PGFLE	CALLED
4	RLFLE	CALLED
3	ERROR-PROCESSING	
4	PRMSE	

SEGMENT NAME: PPGLE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	PPGLE	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
	INPUT-OUTPUT SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	RDCME	
3	PROCESS-NEW-LINE	CALLED
3	CLMSE	
3	UNIT-DATA-LINE	CALLED
3	UNIT-START	
4	SET-LANGUAGE-FLAG	
4	CREATE-INCLUDE-COMMENT	
3	UNIT-END	
3	UNIT-STUB	
4	CREATE-INCLUDE-COMMENT	
3	CREATE-PROJECT-COMMENT	
3	WRITE-SRCLST-PRECOMP	

SEGMENT NAME: PPLKE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	PPLK	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
3	INPUT-OUTPUT SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	STORE-KEYWORDS	
4	INTERPRET-KEYWORD-VALUES	
3	PRINT-FILING-MSG	
4	PRERE	CALLED
3	ASSIGN-OBJECT-FILES	
4	ASFLE	CALLED
3	PROCESS-OBJECT-OPTION	
4	GET-OBJECT-UNIT	
5	FDXEE	CALLED
5	ASFLE	CALLED
4	READ-OBJECT-UNIT-RECORD	
4	WRITE-OBJECT-UNIT-RECORD	
4	READ-OBJECT-UNIT-RECORD	
4	PRERE	CALLED
4	RLFLE	CALLED
4	GENERATE-STUB-UNIT	
5	GET-STUB-OBJECT-UNIT	
6	ASFLE	CALLED
5	READ-OBJECT-UNIT-RECORD	
5	WRITE-OBJECT-UNIT-RECORD	
5	READ-OBJECT-UNIT-RECORD	
5	REPLACE-SYMDEFS	
6	ANBTE	CALLED
6	RECOMPUTE-CHECKSUM	

SEGMENT NAME: PPLKE

UNIT LEVEL	UNIT NAME	UNIT TYPE
5	WRITE-OBJECT-UNIT-RECORD	
5	READ-OBJECT-UNIT-RECORD	
5	PATCH-OBJECT-DECK	
5	WRITE-OBJECT-UNIT-RECORD	
5	PRERE	CALLED
3	PROCESS-LINK-OPTION	
4	GET-LINK-UNIT	
5	ASFLE	CALLED
5	FDXEE	CALLED
5	ITRDE	CALLED
4	RDLNE	CALLED
4	SEARCH-FOR-INCLUDE	
5	PRERE	CALLED
4	GET-OBJECT-UNIT	
5	FDXEE	CALLED
5	ASFLE	CALLED
4	READ-OBJECT-UNIT-RECORD	
4	WRITE-OBJECT-UNIT-RECORD	
4	READ-OBJECT-UNIT-RECORD	
4	RLFLE	CALLED
4	GENERATE-STUB-UNIT	
5	GET-STUB-OBJECT-UNIT	
6	ASFLE	
5	READ-OBJECT-UNIT-RECORD	
5	WRITE-OBJECT-UNIT-RECORD	
5	READ-OBJECT-UNIT-RECORD	
5	REPLACE-SYMDEFS	
6	ANBTE	CALLED
6	RECOMPUTE-CHECKSUM	
5	WRITE-OBJECT-UNIT-RECORD	
5	READ-OBJECT-UNIT-RECORD	
5	PATCH-OBJECT-DECK	

SEGMENT NAME: PPLKE

UNIT LEVEL	UNIT NAME	UNIT TYPE
5	WRITE-OBJECT-UNIT-RECORD	CALLED
5	PRERE	CALLED
4	RDLNE	CALLED
4	RLFLE	CALLED
4	PRERE	CALLED
3	RLAFE	CALLED
3	CLMSE	

SEGMENT NAME: PRAUE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	PRAU	
2	ENVIRONMENT DIVISION	
3	INPUT-OUTPUT SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	STORE-KEYWORD VALUES	
4	OBKWE	CALLED
4	PRMSE	CALLED
4	INTERPRET-VALUE	
5	PRMSE	CALLED
4	PRMSE	CALLED
4	CHECK-FOR-REQUIRED VALUES	
5	PRMSE	CALLED
5	PRMSE	CALLED
5	PRMSE	CALLED
5	PRMSE	CALLED
5	PRMSE	CALLED
3	ASSIGN-SECTION-FILE	
4	ASFLE	CALLED
3	REPORT-PROCESSING	
4	LLINK	CALLED
4	RDXXE	CALLED
4	RDBKE	CALLED
4	LLINK	CALLED
4	PRUNE	CALLED
3	PRINT-INDEX	
4	SETUP-HEADER	
3	RLFLE	CALLED
3	PRMSE	CALLED

SEGMENT NAME: PRCBE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	PRCB	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	RDLNE	CALLED
3	START-NEW-PAGE	
3	START-NEW-PAGE	
3	PREPARE-UNIT-LINE	
4	SETUP-NEXT-WORD	
5	ESTABLISH-WORD-TYPE	
4	SP-RULES-CHECK	
5	SET-UP-NEXT-WORD	
6	ESTABLISH-WORD-TYPE	
4	ADJUST-PRINT-COLUMNS	
5	PROCESS-IF-FIGURE	
6	PUT-FIGURE-IN-STACK	
7	PUT-MSG-IN-LISTING	
8	START-NEW-PAGE	
6	CHECK-FIGURE-IN-STACK	
7	CLOSE-OPEN-FIGURES	
8	PUT-MSG-IN-LISTING	
9	START-NEW-PAGE	
6	PUT-MSG-IN-LISTING	
7	START-NEW-PAGE	
6	CHECK-FIGURE-IN-STACK	
7	CLOSE-OPEN-FIGURES	
8	PUT-MSG-IN-LISTING	
9	START-NEW-PAGE	
5	PROCESS-CASE-FIGURE	
6	PUT-FIGURE-IN-STACK	
7	PUT-MSG-IN-LISTING	
8	START-NEW-PAGE	

SEGMENT NAME: PRCBE

UNIT LEVEL	UNIT NAME	UNIT TYPE
6	CHECK-FIGURE-IN-STACK	
7	CLOSE-OPEN-FIGURES	
8	PUT-MSG-IN-LISTING	
9	START-NEW-PAGE	
6	CHECK-FIGURE-IN-STACK	
7	CLOSE-OPEN-FIGURES	
8	PUT-MSG-IN-LISTING	
9	START-NEW-PAGE	
6	PUT-MSG-IN-LISTING	
7	START-NEW-PAGE	
6	CHECK-FIGURE-IN-STACK	
7	CLOSE-OPEN-FIGURES	
8	PUT-MSG-IN-LISTING	
9	START-NEW-PAGE	
5	PROCESS-DO-FIGURE	
6	PUT-IN-FIGURE-IN-STACK	
7	PUT-MSG-IN-LISTING	
8	START-NEW-PAGE	
6	CHECK-FIGURE-IN-STACK	
7	CLOSE-OPEN-FIGURES	
8	PUT-MSG-IN-LISTING	
9	START-NEW-PAGE	
5	PUT-MSG-IN-LISTING	
6	START-NEW-PAGE	
3	PRINT-LINE	
3	RDLNE	CALLED
3	CLOSE-OPEN-FIGURES	
4	PUT-MSG-IN-LISTING	
5	START-NEW-PAGE	
3	PRINT-INCLUDING-UNITS	
4	START-NEW-PAGE	
4	RDBKE	CALLED
4	START-NEW-PAGE	

SEGMENT NAME: PRFTE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	PRFTE	
2	PRFT-CONFIGURATION-SECTION	
2	PRFT-FILE-SECTION	
2	PRFT-WORKING-STORAGE-77S	
2	PRFT-WORKING-STORAGE-01S	
3	PRFT-FORTRAN-WORD-TABLE	
3	PRFT-INDICES	
3	PRFT-MESSAGES	
3	PRFT-SWITCHES	
2	PRFT-PROCEDURE-DIVISION	
3	PRFT-INITIALIZE-DATA	
3	RDLNE	CALLED
3	PRFT-PRINT-FORTRAN-UNIT	
4	PRFT-COMPRESS-N-SCAN-STATEMENT	
5	PRFT-EVALUATE-PUNCTUATION	
6	PRFT-BYPASS-QUOTED-LITERAL	
6	PRFT-BYPASS-HOLLERITH-FIELD	
4	PRFT-DETERMINE-STATEMENT-TYPE	
5	PRFT-VERIFY-STATEMENT-TYPE	
6	PRFT-VERIFY-TYPE-OF-IF	
4	PRFT-DETERMINE-INDENTATION	
5	PRFT-IF-FIGURE	
5	PRFT-DO-FIGURE	
5	PRFT-CASE-FIGURE	
5	PRFT-BLOCK-FIGURE	
5	PRFT-UNSTRUCTURED-DO	
5	PRFT-END-OF-MODULE	
5	PRFT-END-OF-UNSTRUCTURED-DO	
4	PRFT-CHECK-SP-EXCEPTIONS	
4	PRFT-WRITE-STATEMENT-ON-REPORT	
5	PRFT-MOVE-STATEMENT-TO-LINE	
3	PRFT-PRINT-INCLUDING-UNITS	
4	RDBKE	CALLED

SEGMENT NAME: PRFTE

UNIT LEVEL	UNIT NAME	UNIT TYPE
3	PRFT-CHECK-FIGURE-IN-STACK	
3	PRFT-CLOSE-OPEN-FIGURES	
3	PRFT-GET-STATEMENT	
4	RDLNE	
3	PRFT-PUT-FIGURE-IN-STACK	CALLED
3	PRFT-WRITE-UNIT-LINE	

SEGMENT NAME: PRMDE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	PRMDE	
2	PRMD-ENVIRONMENT-DIVISION	
3	CONFIGURATION SECTION	
3	INPUT-OUTPUT SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
4	PRMD-WORKING-STORAGE-77	
4	PRMD-WORKING-STORAGE-01	
4	PRMD-REPORT-SECTION	
2	PRMD-PROCEDURE-DIVISION	
3	PRMD-STORE-EDIT-KEYWORD-INPUT	
4	PRMSE	CALLED
4	PRMD-INTERPRET-HIST-KW-VALUE	
5	PRMD-CONVERT-HIST-DATE	
5	PRMD-CONVERT-HIST-DATE	
4	PRMSE	CALLED
3	PRMD-INITIATE-MGMT-OPERATIONS	
4	ASFLE	CALLED
4	PRMD-INITIALIZE-READ-MGMT-UNIT	
5	FDXEE	CALLED
5	ITRDE	CALLED
4	PRMD-STORE-FORMAT-ITEM-LABELS	
5	RDLNE	CALLED
5	RDLNE	CALLED
4	PRMSE	CALLED
3	PRMD-INITIATE-DATE-UNIT-INPUT	
4	FDXEE	CALLED
4	RDXXE	CALLED
4	PRMD-FIND-QUALIFYING-ARCHIVE	
4	ITRDE	CALLED
4	RDLNE	CALLED
4	PRMSE	CALLED

SEGMENT NAME: PRMDE

UNIT LEVEL	UNIT NAME	UNIT TYPE
3	PRMD-PROCESS-HEADER-DATA-INPUT	
4	PRMD-SETUP-REPORT-HEADER-LINE	
4	TMRDE	
3	PRMD-PRINT-COLLECTED-DATA-LINE	CALLED
3	RDLNE	
3	PRMSE	CALLED
3	RLAFE	CALLED
		CALLED

SEGMENT NAME: PRM1E

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	PRM1	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
2	DATA DIVISION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	PRINT-TEXT-ONLY-MSGS	
3	PRINT-MSGS-FIRST-10-TEXT	
3	PRINT-MSGS-SPECIAL-TEXT	
4	EXPAND-SECTION-NAME	

SEGMENT NAME: PRM2E

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	PRM2	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
2	DATA DIVISION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	PRINT-MSGS-FIRST-20-TEXT	
4	OBTAIN-UNIT-TYPE	
4	OBTAIN-UNIT-TYPE	
4	EXPAND-SECTION-NAME	
3	PRINT-MSGS-FIRST-50-TEXT	

SEGMENT NAME: PRM3E

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	PRM3	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	PRINT-MSGS-FIRST-35-TEXT	
4	EXPAND-SECTION-NAME	
4	EXPAND-SECTION-NAME	
4	EXPAND-SECTION-NAME	
4	EXPAND-SECTION-NAME	

SEGMENT NAME: PRM4E

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	PRM4	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	PRINT-MULTI-LINES	
4	OBTAIN-UNIT-TYPE	
4	EXPAND-SECTION-NAME	
4	SCAN-FMS-MSG-TEXT	
3	PRINT-MSG-LINES	
4	EXPAND-SECTION-NAME	
4	OBTAIN-UNIT-TYPE	

SEGMENT NAME: PRMRE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	PRMR	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
3	INPUT-OUTPUT SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	STORE-INPUT-KEYWORDS	
4	OBKWE	CALLED
4	INTERPRET-KEYWORD-VALUES	
5	INTERPRET-KEYWORD-LIBRARY	
5	INTERPRET-KEYWORD-PROJECT	
4	PRMSE	CALLED
3	EDIT-INPUT-VALUES	
4	PRMSE	CALLED
4	PRMSE	CALLED
4	PRMSE	CALLED
4	EDIT-PROJECT-LIB-TABLE	
3	WRITE-JCL-FOR-REPORT	
4	ASFLE	CALLED
4	FDXEE	CALLED
4	ITRDE	CALLED
4	PRMSE	CALLED
4	RDLNE	CALLED
4	WRITE-JOB-CARD	
3	WRITE-INPUT-KEYWORD-CARDS	
4	RDLNE	CALLED
3	RLFLE	CALLED
3	PRMSE	CALLED

SEGMENT NAME: PRMSE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	PRMS	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	WRITE-HEADER-LINE	
3	PRINT-MESSAGE-TEXT	
4	LLINK	CALLED
4	PRM1E	CALLED
4	LLINK	CALLED
4	PRM2E	CALLED
4	LLINK	CALLED
4	PRM3E	CALLED
4	LLINK	CALLED
4	PRM4E	CALLED
4	WRITE-LINE	
4	WRITE-LINE	

SEGMENT NAME: PRMSI

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	PRMSI	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	PRERE	CALLED
3	INITIALIZE-PRINT-LINE	
3	PRINT-MULTI-LINES	
4	SEARCH-UNIT-TABLE	
4	EXPAND-SECTION-NAME	
4	SCAN-FMS-MSG-TEXT	
4	PRINT-MSG-LINES	
5	EXPAND-SECTION-NAME	
3	PRINT-TEXT-ONLY-MSGS	
3	PRINT-MSGS-FIRST-10-TEXT	
3	PRINT-MSGS-FIRST-20-TEXT	
4	SEARCH-UNIT-TABLE	
3	PRINT-MSGS-FIRST-35-TEXT	
4	EXPAND-SECTION-NAME	
4	EXPAND-SECTION-NAME	
3	PRINT-MSGS-FIRST-50-TEXT	
3	PRINT-MSGS-SPECIAL-TEXT	
4	EXPAND-SECTION-NAME	

SEGMENT NAME: PRNIE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	PRNIE	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
3	INPUT-OUTPUT SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	RDLNE	CALLED
3	START-NEW-PAGE	
3	START-NEW-PAGE	
3	RDLNE	CALLED
3	PRINT-INCLUDING-UNITS	
4	START-NEW-PAGE	
4	RDBKE	CALLED
4	START-NEW-PAGE	

SEGMENT NAME: PRPSE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	PRPSE	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
3	INPUT-OUTPUT SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
3	REPORT SECTION	
2	PROCEDURE DIVISION	
3	DATE-ROUTINE	
3	CHECK-KEYWORDS	
4	ERROR-PROCESSING	
5	PRERE	CALLED
4	ERROR-PROCESSING	
5	PRERE	CALLED
4	ERROR-PROCESSING	
5	PRERE	CALLED
3	RDPSE	CALLED
3	DATA-LINE-PROCESSING	
4	WRITE-PS-DATA-LINE	
5	CONVERT-UNIT-TYPE-CODE	
5	DATE-ROUTINE	
5	DATE-ROUTINE	
4	SETUP-XREF-FILE	
4	SETUP-CALL-TABLE	
5	ERROR-PROCESSING	
6	PRERE	CALLED
4	RDPSE	CALLED
3	PRINT-XREF-LISTING	
3	CLMGE	CALLED

SEGMENT NAME: PRSDE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	PRSDE	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
3	INPUT-OUTPUT SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	STORE-KEYWORD-VALUE	
4	OBKWE	CALLED
4	PRMSE	CALLED
4	INTERPRET-VALUE	
5	ERROR-PROCESSING	
6	PRMSE	CALLED
5	ERROR-PROCESSING	
6	PRMSE	CALLED
5	ERROR-PROCESSING	
6	PRMSE	CALLED
5	ERROR-PROCESSING	
6	PRMSE	CALLED
5	ERROR-PROCESSING	
6	PRMSE	CALLED
5	ERROR-PROCESSING	
6	PRMSE	CALLED
5	ERROR-PROCESSING	
6	PRMSE	CALLED
5	ERROR-PROCESSING	
6	PRMSE	CALLED

SEGMENT NAME: PRSDE

UNIT LEVEL	UNIT NAME	UNIT TYPE
4	CHECK-FOR-REQUIRED-VALUES	
5	PRMSE	CALLED
5	PRMSE	CALLED
5	PRMSE	CALLED
5	PRMSE	CALLED
5	PRMSE	CALLED
5	PRMSE	CALLED
3	ASSIGN-SECTION-FILE	
4	ASFLE	CALLED
3	GET-INDEX-ENTRY	
4	LLINK	CALLED
4	RDXXE	CALLED
4	FDXEE	CALLED
3	SETUP-UNIT-FILE-NAME	
3	LLINK	CALLED
3	OUTPUT-UNIT	
4	PRUNE	CALLED
4	PUNCH-UNIT	
5	ITRDE	CALLED
5	RDLNE	CALLED
5	RDLNE	CALLED
4	WRITE-UNIT-TO-TAPE	
5	ITRDE	CALLED
5	RDLNE	CALLED
5	RDLNE	CALLED
3	RDXXE	CALLED
3	SET-UNIT-FILE-NAME	
3	OUTPUT-UNIT	
4	PRUNE	CALLED
4	PUNCH-UNIT	

SEGMENT NAME: PRSDE

UNIT LEVEL	UNIT NAME	UNIT TYPE
5	ITRDE	CALLED
5	RDLNE	CALLED
5	RDLNE	CALLED
4	WRITE-UNIT-TO-TAPE	
5	ITRDE	CALLED
5	RDLNE	CALLED
5	RDLNE	CALLED
3	PRMSE	CALLED
3	RLFLE	CALLED

SEGMENT NAME: PRUNE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	PRUNE	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
2	DATA DIVISION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	ITRDE	CALLED
3	SETUP-HEADER	
4	SETUP-HEADER-UNIT-TYPE	
3	ESTABLISH-PRINT-FORMAT	
3	LLINK	CALLED
3	PRNIE	CALLED
3	PRNIE	CALLED
3	PRNIE	CALLED
3	PRNIE	CALLED
3	PRNIE	CALLED
3	PRPLE	CALLED
3	PRMSE	CALLED

A-88

SEGMENT NAME: PRXXE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	PRXXE	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
3	INPUT-OUTPUT SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	STORE-KEYWORD-VALUES	
4	OBKWE	CALLED
4	PRMSE	CALLED
4	INTERPRET-VALUE	
5	ERROR-PROCESSING	
6	PRMSE	CALLED
5	ERROR-PROCESSING	
6	PRMSE	CALLED
5	ERROR-PROCESSING	
6	PRMSE	CALLED
5	ERROR-PROCESSING	
6	PRMSE	CALLED
5	ERROR-PROCESSING	
6	PRMSE	CALLED
4	CHECK-FOR-REQUIRED-VALUES	
5	PRMSE	CALLED
5	PRMSE	CALLED
5	PRMSE	CALLED
3	ASSIGN-SECTION-FILE	
4	ASFLE	CALLED
3	LLINK	CALLED
3	SETUP-HEADER	
4	CALCULATE-BLOCKS-USED	
5	RDBKE	CALLED

SEGMENT NAME: PRXXE

UNIT LEVEL	UNIT NAME	UNIT TYPE
5	SCAN-BITS-IN-CHAR	
5	SCAN-BITS-IN-CHAR	
3	GET-UNIT-DATA-BLOCK	
4	RDXXE	CALLED
4	RDBKE	CALLED
3	START-NEW-PAGE	
3	START-NEW-PAGE	
3	SETUP-LISTING-LINE	
3	PRMSE	CALLED
3	RLFLE	CALLED

SEGMENT NAME: RDCME

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	RDCM	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
3	INPUT-OUTPUT SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	INITIALIZE-MODULE	
4	INITIALIZE-LIBRARY	
5	ASFLE	CALLED
5	PRMSE	CALLED
4	PRINT-FILING-MSG	
5	ERROR-PROCESSING	
6	PRERE	CALLED
4	FIND-UNIT-IN-LIBRARIES	
5	FDXEE	CALLED
5	ERROR-PROCESSING	
6	PRERE	CALLED
5	ERROR-PROCESSING	
6	PRERE	CALLED
4	INITIALIZE-UNIT	
5	ITRDE	CALLED
5	IMRDE	CALLED
5	ERROR-PROCESSING	
6	PRERE	CALLED
3	RDLNE	CALLED
3	PROCESS-INCLUDE	
4	FIND-INCLUDE	
4	ERROR-PROCESSING	
5	PRERE	CALLED
4	ERROR-PROCESSING	
5	PRERE	CALLED
4	FIND-UNIT-IN-LIBRARIES	

SEGMENT NAME: RDCME

UNIT LEVEL	UNIT NAME	UNIT TYPE
5	ERROR-PROCESSING	
6	PRERE	CALLED
5	ERROR-PROCESSING	
6	PRERE	CALLED
4	INITIALIZE-UNIT	
5	ITRDE	CALLED
5	TMRDE	CALLED
5	ERROR-PROCESSING	
6	PRERE	CALLED
4	ERROR-PROCESSING	
5	PRERE	CALLED
3	ERROR-PROCESSING	
4	PRERE	CALLED
3	RLAFE	CALLED
3	TERMINATE-UNIT	
3	TERMINATE-MODULE	
4	RLAFE	CALLED

SEGMENT NAME: RDPSE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	RDPS	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
3	INPUT-OUTPUT	
2	DATA-DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	INITIALIZE-MODULE	
4	EDIT-INPUT-VALUES	
5	PRMSE	CALLED
5	PRMSE	CALLED
5	PRMSE	CALLED
5	PRMSE	CALLED
4	INITIALIZE-LIBRARY	
5	ASFLE	CALLED
5	PRMSE	CALLED
4	FIND-UNIT-IN-LIBRARIES	
5	FDXEE	CALLED
5	ERROR-PROCESSING	
6	PRERE	CALLED
5	ERROR-PROCESSING	
6	PRERE	CALLED
4	INITIALIZE-UNIT	
5	ITRDE	CALLED
5	ERROR-PROCESSING	
6	PRERE	CALLED
3	RDLNE	CALLED
3	PROCESS-INCLUDE-AND-CALL	
4	FIND-INCLUDE-AND-CALL	
4	ERROR-PROCESSING	
5	PRERE	CALLED
4	ERROR-PROCESSING	
5	PRERE	CALLED

SEGMENT NAME: RDPSE

UNIT LEVEL	UNIT NAME	UNIT TYPE
4	FIND-UNIT-IN-LIBRARIES	
5	FDXEE	CALLED
5	ERROR-PROCESSING	
6	PRERE	CALLED
5	ERROR-PROCESSING	
6	PRERE	CALLED
4	ITRDE	CALLED
4	TMRDE	
4	ERROR-PROCESSING	
5	PRERE	CALLED
4	INITIALIZE-UNIT	
5	ITRDE	
5	ERROR-PROCESSING	
6	PRERE	CALLED
4	ERROR-PROCESSING	
5	PRERE	CALLED
3	ERROR-PROCESSING	
4	PRERE	CALLED
3	RLAFE	CALLED
3	TERMINATE-UNIT	
3	TERMINATE-MODULE	CALLED
4	RLAFE	

SEGMENT NAME: RDXXE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	RDXX	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
2	DATA DIVISION	
3	WORKING STORAGE SECTION	
2	PROCEDURE DIVISION	
3	INITIALIZE-INDEX-BLOCK	
4	RDBKE	CALLED
4	RDBKE	CALLED
3	GET-BLOCK-FROM-STACK	
4	RDBKE	CALLED
3	PUT-BLOCK-INTO-STACK	
4	RDBKE	CALLED
3	ERROR-PROCESSING	
4	PRMSE	CALLED

SEGMENT NAME: RPUNE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	RRUN	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
2	DATA DIVISION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	LLINK	CALLED
3	STORE-REPLACE-KEYWORDS	
4	OBKWE	CALLED
4	PRMSE	CALLED
4	INTERPRET-KEYWORDS	
4	PRMSE	CALLED
4	EDIT-INPUT-VALUES	
5	PRMSE	CALLED
5	PRMSE	CALLED
5	PRMSE	CALLED
5	PRMSE	CALLED
5	PRMSE	CALLED
3	LOCATE-UNIT	
4	ASFLE	CALLED
4	FDXEE	CALLED
4	PRMSE	CALLED
3	DETERMINE-UNIT-TYPE	
3	DELETE-UNIT	
4	ITRDE	CALLED
4	PRMSE	CALLED
4	TMRDE	CALLED
4	DLUNE	CALLED
3	REPLACE-UNIT-DATA	
4	ITWRE	CALLED
4	LLINK	CALLED
4	OBSDE	CALLED

SEGMENT NAME: RPUNE

UNIT LEVEL	UNIT NAME	UNIT TYPE
4	STORE-LOWER-UNIT-INFO	
5	PCLUE	CALLED
5	PRMSE	CALLED
4	WRLNE	CALLED
4	PCLUE	CALLED
4	OBSDE	CALLED
4	TMWRE	CALLED
3	UPDATE-ACCOUNT-INFO	
3	UPDATE-UNIT-INDEX	
4	ADXEE	CALLED
4	CHXEE	CALLED
4	WRACE	CALLED
3	PRINT-NEW-UNIT	
4	LLINK	CALLED
4	PRUNE	CALLED
4	WRACE	CALLED
3	TERMINATE-REPLACE	
4	RLFLE	CALLED
4	OBSDE	CALLED
4	PRMSE	CALLED

SEGMENT NAME: RSLBE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	RSLB	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
3	INPUT-OUTPUT SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	STORE-KEYWORDS	
4	LLINK	CALLED
4	OBKWE	CALLED
4	PRMSE	CALLED
4	INTERPRET-KEYWORD-VALUES	
4	PRMSE	CALLED
3	EDIT-KEYWORDS	
4	PRMSE	CALLED
4	PRMSE	CALLED
4	PRMSE	CALLED
4	PRMSE	CALLED
4	PRMSE	CALLED
4	PRMSE	CALLED
4	PRMSE	CALLED
3	RESTORE-LIBRARY-UNITS	
4	RECORD-SEQUENCE-ERROR	
5	PRMSE	CALLED
5	UNIT-WINDUP	
6	TMWRE	CALLED
6	DAFLE	CALLED
6	LLINK	CALLED
6	PRUNE	CALLED
6	PRMSE	CALLED
5	SECTION-WINDUP	

SEGMENT NAME: RSLBE

UNIT LEVEL	UNIT NAME	UNIT TYPE
6	RLFLE	CALLED
6	PRMSE	CALLED
5	PRMSE	CALLED
5	UNIT-WINDUP	
6	DAFLE	CALLED
6	LLINK	CALLED
6	PRUNE	CALLED
6	PRMSE	CALLED
4	SECTION-WINDUP	
5	RLFLE	CALLED
5	PRMSE	CALLED
4	LIB-SEC-VERIFICATION	
5	ASFLE	CALLED
5	RDBKE	CALLED
5	WRBKE	CALLED
5	RLBKE	CALLED
5	RLFLE	CALLED
5	PRMSE	CALLED
5	PRMSE	CALLED
4	PSL-UNIT-INITIATION	
5	UNIT-INDEX-CHECK	
6	FDXEE	CALLED
6	RDBKE	CALLED
5	UNIT-DELETION	
6	LLINK	CALLED
6	DLUNE	CALLED
6	PRMSE	CALLED
6	UNIT-INDEX-CHECK	
7	FDXEE	CALLED
7	RDBKE	CALLED
6	PRMSE	CALLED
5	WRITE-ACCOUTING-INFO	
6	PRMSE	CALLED

SEGMENT NAME: RSLBE

UNIT LEVEL	UNIT NAME	UNIT TYPE
6	WRACE	CALLED
6	ITWRE	CALLED
5	CHXEE	CALLED
5	ADXEE	CALLED
5	LLINK	CALLED
4	DEPENDENT-UNIT-RESTORE	
5	STORE-LOWER-UNIT-INFO	
6	PCLUE	CALLED
6	PRMSE	CALLED
5	WRLNE	CALLED
5	STORE-LOWER-UNIT-INFO	
6	PCLUE	CALLED
6	PRMSE	CALLED
4	INDEPENDENT-FILE-CREATE	
5	STCFE	CALLED
5	CRFLE	CALLED
5	ALFLE	CALLED
4	INDEPENDENT-UNIT-RESTORE	
5	PRMSE	
4	UNIT-WINDUP	
5	TMWRE	CALLED
5	DAFLE	CALLED
5	LLINK	CALLED
5	PRUNE	CALLED
5	PRMSE	CALLED
4	SECTION-WINDUP	
5	RLFLE	CALLED
5	PRMSE	CALLED
4	FORMAT-ERROA-RESOLUTION	
5	PRMSE	CALLED
3	PRMSE	CALLED
3	PRMSE	CALLED
3	PRMSE	CALLED

SEGMENT NAME: SCSGE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	SCSGE	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
3	INPUT-OUTPUT SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	STORE-KEYWORD-VALUES	
4	OBKWE	CALLED
4	PRMSE	CALLED
4	INTERPRET-VALUE	
4	PRMSE	CALLED
4	CHECK-FOR-REQUIRED-VALUES	
5	PRMSE	CALLED
5	PRMSE	CALLED
5	PRMSE	CALLED
5	PRMSE	CALLED
5	PRMSE	CALLED
5	PRMSE	CALLED
3	ASSIGN-SECTION-FILE	
4	ASFLE	CALLED
3	LLINK	CALLED
3	SET-UP-HEADER	
3	START-NEW-PAGE	
3	RDXXE	CALLED
3	SETUP-UNIT-FILE-NAME	
3	SCAN-UNIT-BY-LINE	
4	START-NEW-PAGE	
4	ITRDE	CALLED
4	RDLNE	CALLED

SEGMENT NAME: SCSGE

UNIT LEVEL	UNIT NAME	UNIT TYPE
4	SCAN-LINE-FOR-STG-VALUE	
5	START-NEW-PAGE	
5	START-NEW-PAGE	
3	PRMSE	CALLED
3	RLFLE	CALLED

SEGMENT NAME: STCFE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	STCFE	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
3	INPUT-OUTPUT SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	CALLED
3	DEFAULT-VALUES	
3	LLINK	CALLED
3	PROCESS-FMS-DIRECTIVES	
4	PRMSE	
4	INTERPRET-USER-OPTIONS	
5	PROCESS-ABORT	
5	PROCESS-ACCESS	
5	PROCESS-PERMISSIONS	
6	LOCATE-SPECIFIC-USER	
7	FILL-PERMISSION-LIST	
8	ORBTE	CALLED
8	READ-FOR-VALUE	
7	PRMSE	CALLED
6	PRMSE	CALLED
6	ORBTE	CALLED
5	PROCESS-AUDIT	
5	PROCESS-FILE-SPACE	
6	NUMERIC-EDIT	
5	PROCESS-DEVICES	
6	PRMSE	CALLED
6	PRMSE	CALLED
5	PROCESS-EXCLUDES	
6	LOCATE-SPECIFIC-USER	
7	FILL-PERMISSION-LIST	
8	ORBTE	CALLED
8	READ-FOR-VALUE	

SEGMENT NAME: STCFE

UNIT LEVEL	UNIT NAME	UNIT TYPE
7	PRMSE	CALLED
6	PRMSE	CALLED
6	ORBTE	CALLED
5	PROCESS-FCLASS	
5	PROCESS-INCRSAVE	
5	PROCESS-PAGESIZE	
6	NUMERIC-EDIT	
5	PROCESS-RDERR	
6	PROCESS-RDERR-DEVICE	
7	PRMSE	CALLED
5	PROCESS-VERIFY	
4	PRMSE	CALLED
3	QUERY-FOR-CLASS	
4	LLINK	CALLED
4	QYSCE	CALLED
3	PRMSE	CALLED

SEGMENT NAME: STMFE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	STMFE	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
3	INPUT-OUTPUT SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	SETUP-DEFAULT-VALUES	
3	LLINK	CALLED
3	PROCESS-FMS-DIRECTIVES	
4	PRMSE	CALLED
4	INTERPRET-USER-OPTIONS	
5	ANBTE	CALLED
5	PROCESS-ABORT	
5	PROCESS-ACCESS	
5	PROCESS-PERMISSIONS	
6	LOCATE-SPECIFIC-USER	
7	FILL-PERMISSION-LIST	
8	ORBTE	CALLED
8	READ-FOR-VALUE	
6	PRMSE	CALLED
6	ORBTE	CALLED
5	PROCESS-AUDIT	
5	PROCESS-FILE-SPACE	
6	NUMERIC-EDIT	
5	PROCESS-DELETE	
6	READ-FOR-VALUE	
6	PRMSE	CALLED
5	PROCESS-EXCLUDES	
6	LOCATE-SPECIFIC-USER	
7	FILL-PERMISSION-LIST	
8	ORBTE	CALLED
8	READ-FOR-VALUE	

SEGMENT NAME: STMFE

UNIT LEVEL	UNIT NAME	UNIT TYPE
5	PROCESS-FCLASS	
5	PROCESS-INCRSAVE	
5	PROCESS-PAGESIZE	
6	NUMERIC-EDIT	
5	PROCESS-RDERR	
6	PROCESS-RDERR-DEVICE	
7	PRMSE	
5	PROCESS-RESET	CALLED
5	PROCESS-VERIFY	
4	PRMSE	
3		CALLED

SEGMENT NAME: UPCLE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	UPCLE	
2	CLMD-ENVIRONMENT-DIVISION	
3	CONFIGURATION SECTION	
3	INPUT-OUTPUT SECTION	
2	DATA DIVISION	
3	CLMD-FILE-SECTION	
3	CLMD-WORKING-STORAGE-77	
3	CLMD-WORKING-STORAGE-01	
2	UPCL-PROCEDURE-DIVISION	
3	CLMD-ASSIGN-MGMT-SECTION-FILE	
4	ASFLE	CALLED
3	CLMD-STORE-ARCHIVE-COLLECTION	
4	CLMD-ASSIGN-MGMT-SECTION-FILE	
5	ASFLE	CALLED
4	CLMD-INITIALIZE-NEW-COLLECTION	
5	INITIALIZE-READ-MGMT-UNIT	
6	FDXEE	CALLED
6	ITRDE	CALLED
6	PRMSE	CALLED
5	TMRDE	CALLED
5	ITRDE	CALLED
5	ITWRE	CALLED
5	ADXEE	CALLED
5	RLBKE	CALLED
4	WRLNE	CALLED
4	TMWRE	CALLED
4	CLMD-ARCHIVE-PAST-COLLECTION	
5	INITIALIZE-READ-MGMT-UNIT	
6	FDXEE	CALLED
6	ITRDE	CALLED
6	PRMSE	CALLED
5	ADXEE	CALLED

SEGMENT NAME: UPCLE

UNIT LEVEL	UNIT NAME	UNIT TYPE
5	WRACE	CALLED
5	DLXEE	CALLED
5	CHXEE	CALLED
4	CHXEE	CALLED
4	RLBKE	CALLED
4	RLBKE	CALLED
4	DLXEE	CALLED
4	WRACE	CALLED
3	RLFLE	CALLED

SEGMENT NAME: UPMDE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	UPMDE	CALLED
2	UPMD-CONFIGURATION-SECTION	
2	UPMD-FILE-SECTION	
2	UPMD-WORKING-STORAGE-SECTION	
3	CLIB-CODES-FOR-UPMD	
3	CLIB-LINK-NAMES-FOR-UPMD	
2	UPMD-PROCEDURE-DIVISION	
3	LLINK	CALLED
3	UPMFE	CALLED
3	LLINK	CALLED
3	LLINK	CALLED
3	UPMRE	CALLED
3	UPMDE-CONCLUDE-MDUPDATE	
4	RLBKE	CALLED
4	OBSDE	CALLED
4	PRMSE	CALLED
4	LLINK	CALLED
4	PRUNE	CALLED
4	RLFLE	CALLED

SEGMENT NAME: UPMFE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	UPMFE	CALLED
2	UPMFE-CONFIGURATION-SECTION	
2	UPMFE-FILE-SECTION	
2	UPMFE-WORKING-STORAGE-77S	
2	UPMFE-WORKING-STORAGE-01S	
3	CLIB-CODES-FOR-UPMD	
3	CLIB-MDCR-DATA-RECORD-FORMAT	
3	CLIB-LINK-NAMES-FOR-UPMD	
3	CLIB-MDCR-UNIT-NAMES	
3	CLIB-UPMD-FUNCTION-VALUES	
2	UPMFE-PROCEDURE-DIVISION	
3	UPMD-STORE-KEYWORDS	
4	OBFNE	CALLED
4	OBKWE	CALLED
4	PRMSE	CALLED
4	UPMD-INTERPRET-PARAM-VALUES	
4	UPMD-INTERPRET-UPDATE-VALUES	
5	PRMSE	CALLED
4	PRMSE	CALLED
4	UPMD-EDIT-INPUT-VALUES	
5	PRMSE	CALLED
5	PRMSE	CALLED
5	PRMSE	CALLED
5	PRMSE	CALLED
5	PRMSE	CALLED
5	PRMSE	CALLED
3	UPMD-LOCATE-MGMT-DATA	
4	ASFLE	CALLED
4	FDXEE	CALLED
4	PRMSE	CALLED
4	PRMSE	CALLED
3	UPMD-OBTAIN-FORMAT-DATA	
4	ITRDE	CALLED

SEGMENT NAME: UPMFE

UNIT LEVEL	UNIT NAME	UNIT TYPE
4	UPMFE-CHECK-ACCT-INFO	
5	PRMSE	CALLED
5	LLINK	CALLED
5	PRUNE	CALLED
5	PRMSE	CALLED
4	[illegible]DE	CALLED
4	PRMSE	CALLED
4	PRXEE	CALLED
4	[illegible]-FORMAT-TABLE	
5	IENDE	CALLED
5	ENDLSE	CALLED
5	PRMSE	CALLED
5	[illegible]DE	CALLED
4	PRMSE	CALLED
3	UPMD-VERIFY-AND-SORT-DATA	
3	UPMF-SEARCH-[illegible]AT-TABLE	
4	PRMSE	CALLED
4	PRMSE	CALLED
3	UPMF-[illegible]MERIC-EDIT	
3	UPMF-OBTAIN-VALUE	
3	UPMF-OBTAIN-ITEM-NBRS	
4	PRMSE	CALLED
3	UPMD-OBTAIN-USER-DATA	
4	OBSDE	CALLED
4	UPMF-VERIFY-ITEM-NBRS	
5	PRMSE	CALLED
5	PRMSE	CALLED
4	UPMD-EDIT-XCHECK-DATA	
5	UPMF-EDIT-VARIANCE	
6	PRMSE	CALLED
5	UPMF-EDIT-DATE	
6	PRMSE	CALLED

SEGMENT NAME: UPMFE

UNIT LEVEL	UNIT NAME	UNIT TYPE
5	UPMF-EDIT-REMARK	
6	PRMSE	
4	UPMD-EDIT-REGULAR-DATA	CALLED
5	UPMF-ALPHA-NUMERIC-DATA-EDIT	
5	PRMSE	
3	UPMD-GET-SUBFUNCTION	CALLED
4	OBFNE	
4	OBKWE	CALLED
		CALLED

SEGMENT NAME: UPMRE

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	UPMRE	CALLED
2	UPMF-CONFIGURATION-SECTION	
2	UPMF-FILE-SECTION	
3	CLIB-MDCR-DATA-RECORD-FORMAT	
2	UPMF-WORKING-STORAGE-77S	
2	UPMF-WORKING-STORAGE-01S	
3	CLIB-CODES-FOR-UPMD	
3	CLIB-MDCR-DATA-RECORD-FORMAT	
3	CLIB-LINK-NAMES-FOR-UPMD	
2	UPMF-PROCEDURE-DIVISION	
3	ITWRE	CALLED
3	UPMF-ADD-DATA	
4	WRLNE	CALLED
4	TMWRE	CALLED
4	ADXEE	CALLED
4	UPMF-ADD-ACCT-INFO	
5	WRACE	CALLED
3	UPMF-CHANGE-DATA	
4	ITRDE	CALLED
4	RDLNE	CALLED
4	WRLNE	CALLED
4	RDLNE	CALLED
4	UPMF-CHANGE-DATA-RECORDS	
5	UPMR-CHANGE-LINE	
6	WRLNE	CALLED
6	PRMSE	CALLED
6	WRLNE	CALLED
6	RDLNE	CALLED
5	WRLNE	CALLED
5	PRMSE	CALLED
5	WRLNE	CALLED
5	RDLNE	CALLED
4	UPMR-LAST-INSERTS	

SEGMENT NAME: UPMRE

UNIT LEVEL	UNIT NAME	UNIT TYPE
5	WRLNE	CALLED
5	PRMSE	CALLED
4	TMWRE	CALLED
4	TMRDE	CALLED
4	CHXEE	CALLED
4	RLBKE	CALLED
4	UPMF-CHANGE-ACCT-INFO	
5	WRACE	CALLED
3	UPMF-DELETE-DATA	
4	ITRDE	CALLED
4	PRMSE	CALLED
4	TMRDE	CALLED
4	DLXEE	CALLED
4	RLBKE	CALLED

SEGMENT NAME: WRAC

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	WRAC	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
2	DATA DIVISION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	FDXXE	
3	RDBKE	CALLED
3	ASBKE	CALLED
3	WRBKE	CALLED
3	PRMSE	CALLED
		CALLED

SEGMENT NAME: WRJB

UNIT LEVEL	UNIT NAME	UNIT TYPE
1	WRJBE	
2	ENVIRONMENT DIVISION	
3	CONFIGURATION SECTION	
3	INPUT-OUTPUT SECTION	
2	DATA DIVISION	
3	FILE SECTION	
3	WORKING-STORAGE SECTION	
2	PROCEDURE DIVISION	
3	OBKWE	CALLED
3	OBSDE	CALLED
3	OBSDE	CALLED
3	PRMSE	CALLED

APPENDIX B. LIBRARY PSLPRG LINKS

```
I**.ZPSL(1).PPJVMAP
      1       IN S.PPJVE
      2       IN S.ALLOCATOR
      *       IN S.ACPMP..ASBKE..RDBKE..WRBKE..ACFLP..ASFLE
      4       IN S.PLAFE..PLFLE..TYPDE
      *       IN S.PPFITEM..ALFLE..ERCSF..RDCME
      6       IN S.DAFLF..CMFLE..SCJBF..CRFLE..OYFIE
      7       IN S.PRMSI..FDVSE..PRFRE
      *       IN S.ITPDE..PDLNE..ITPDE1..ALPKE
      *       E*D
```

```
IBM•ZPSL(1).PREC•MAP
      1       IN S.PRECOMPILER
      2       IN S.PRERE
      3       FWD
```

```
DMATC$PSL.IPM-PSL(1).PCTLMAP
     1        IN S.PCTLE
     2        IN S.WRACE..CMBEE..ITWRE..ITWRE1..WRLNE
     3        IN S.ADVEE..ACEWE..ASEWE..RDBKE..WRBKE..ACFLE..ASFLE
     4        IN S.FLAEE..RLFLE..DLYEE..RDYXE..STUBS1
     5        IN S.PRICE..CEENE..OPNWE..OPSDE..FLFLE..ALLOCATOR..ERCSE
     6        IN S.PRMSE..FDYEE..CPUSE..TMWRE
     7        IN S.ITPDE..RELNE..ITRDS1..RLBKE..TMRDE
     8        IN S.DAFLE..SEJBE..CBFLE..CMFLE..FGFLE
     9        IN S.RFITEM..CYBIE
    10        SEG LKPRXX.
    11        IN S.PRXXE
    12        SEG LKESPA..LKPRXX
    13        IN S.ESPAE
    14        SEG LKITPJ..LKPEXX
    15        IN S.ITPJE
    16        SEG LKPRSC..LKPRXX
    17        IN S.PRSCE
    18        SEG LKCMML..LKPEXX
    19        IN S.CMMLE
    20        SEG LKEXPG..LKPEXX
    21        IN S.EXPGE
    22        SEG LKLKPG..LKPEXX
    23        IN S.LKPGE
    24        SEG LKITCL..LKPRXX
    25        IN S.ITCLE
    26        SEG LKPRMR..LKPRXX
    27        IN S.PRMRE
    28        SEG LKPRAU..LKPRXX
    29        IN S.PRAUE
    30        SEG LKPEJB..LKPEXX
    31        IN S.PEJBE
    32        SEG LKPCML..LKPRXX
    33        IN S.PCMLE
    34        SEG LKSCSC..LKPEXX
    35        IN S.SCSGE
    36        SEG LKPRDE..LKPRXX
    37        IN S.PRDCE
    38        SEG LKWRJB..LKPRXX
    39        IN S.PRJBE
    40        SEG LKRPUN..LKPEXX
    41        IN S.RPUNE
    42        SEG LKPGUN..LKPEXX
    43        IN S.PGUNE
    44        SEG LKADUN..LKPEXX
    45        IN S.ADUNE
    46        SEG LKCRSC..LKPEXX
    47        IN S.CRSCE
    48        SEG LKCHUN..LKPRXX
    49        IN S.CHUNE
    50        SEG LKPMMD..LKPRXX
    51        IN S.PMMDE
    52        SEG LKUPMD..LKPRXX
    53        IN S.UPMDE
    54        SEG LKMVUN..LKPEXX
    55        IN S.MVUNE
    56        SEG LKLBL..LKLBXX
```

```
57        IN S.FKLRE
58        SEG LKPGSC*,LKPRXX
59        IN S.PGSCE
60        SEG LKRSLE*,LKPPXX
61        IN S.RSLRE
62        SEG LKLLNK*,LKPRXX
63        IN S.LLINK1
64        SEG LKPRM1*,()
65        IN S.PRM1E
66        SEG LKPRM2*,LKPRM1
67        IN S.PRM2E
68        SEG LKPRM3*,LKPRM1
69        IN S.PRM3E
70        SEG LKPPM4*,LKPRM1
71        IN S.PRM4E
72        SEG LKCHLN*,()
73        IN S.CHLNE
74        SEG LKCHRC*,LKCHLN
75        IN S.CHRCE
76        SEG LKFMED*,LKCHLN
77        IN S.FMEDE
78        SEG LKFMVE*,LKCHLN
79        IN S.FMVE
80        SEG LKFMUP*,LKCHLN
81        IN S.FMUPE
82        SEG LKUMPR*,LKCHLN
83        IN S.UPMRE
84        SEG LKUPME*,LKCHLN
85        IN S.UPMEE
86        SEG LKBKSC*,LKCHLN
87        IN S.FKSCE
88        SEG LKPPLK*,LKCHLN
89        IN S.PPLKE
90        SEG LKITSF*,()
91        IN S.ITSFE
92        SEG LKDLUN*,LKITSF
93        IN S.PLUNE
94        SEG LKPCLU*
95        IN S.PCLUE
96        SEG LKPRUN*,LKITSF
97        IN S.PRUNE
98        SEG LKPRNI*
99        IN S.PRNIE
100       SEG LKPRCE*,LKPRNI
101       IN S.PRCPE
102       SEG LKPPFT*,LKPRNI
103       IN S.PRFTE
104       SEG LKPRJV*,LKPRNI
105       IN S.PRJVE
106       END
```

```
DMATCSPSL-IBP-PSL(1).PPCPMAP
      1       IN S.FPCFE..ALLOCATOR
      2       IN S.ACPKE..ASBKE..RDPKE..WPBKE..ACFLE..ASFLE
      3       IN S.RLAFE..PLFLE..TYRDE
      4       IN S.FRFITEM..ALFLE..FRCSF..RDCMF
      5       IN S.CAFLE..CMFLE..SPJBE..CRFLE..GYFIF
      6       IN S.FRMSI..FDXFF..PRFRE
      7       IN S.ITRDE..RDLNE..ITRDE1..RLFKE
      8       END
```

```
DMATC1PSL*IPM-PSL(1).PPGLMAP
1       IN S.PPGLE
2       IN S.ALLOCATOR
3       IN S.ACBKE,,ASBKE,,RDEKE,,WFBKE,,ACFLE,,ASFLE
4       IN S.PLAFE,,PLFLE,,TMRDE
5       IN S.ERFITEM,,ALFLE,,ERCSF,,RDCME
6       IN S.DBFLE,,CMFLE,,SPJPE,,CPFLE,,CYFIE
7       IN S.FRMSI,,FDXEE,,PPERE
8       IN S.ITRDE,,RDLNF,,ITRDE1,,PLEKE
9       END
```

```
DMATC$PSL.IPM-PSL(1).PPFTMAP
      1       IN S.MAIN..BLOCK
      2       LIB C.
      3       END
```

```
DPATC$PSL.IBM-PSL(1).PRPSMAP
      1       IN S.PRPSE..PDPSE
      2       IN S.ACBKE..ASBKE..RCBKE..WRBKE..ACFLE..ASFLE
      3       IN S.RLAFE..FLFLE..TWRDE
      4       IN S.ERFITEM..ALFLE..ERCSF
      5       IN S.ALLOCATOR
      6       IN S.DAFLE..CHFLE..SPJBE..CRFLE..CYFIE
      7       IN S.PRMSI..FDXEE..PRERE
      8       IN S.ITRDE..RDLNE..ITRDE1..RLBKE
      9       END
```

```
DMATC$PSL.IBM-PSL(1).CLMDMAP
     1        IN S.CLMDF
     2        IN S.ALLOCATOR
     3        IN S.ACRKF,,ASRKF,,PDRKF,,WRRKF,,ACFL9,,ASFLE
     4        IN S.PLAFE,,PLFLE,,TMRDF,,TMWRF
     5        IN S.ERFITEM,,ALFLE,,ERCSF
     6        IN S.DAFLE,,CMFLE,,SPJBF,,CRFLE,,QYFIE
     7        IN S.PRMSF,,FDXEE
     8        IN S.CMXEF,,ADXEE,,DLXEF,,RDXXE
     9        IN S.ITWRE,,WRLME,,ITWRE1,,STURS1
    10        IN S.ITRDE,,RDLME,,ITRDE1,,RLEKE,,WRACE
    11        SEG LKPPCL.
    12        IN S.PPCLE
    13        SEG LKPCCL,,LKPPCL
    14        IN S.PCCLE
    15        SEG LKUPCL,,LKPPCL
    16        IN S.UPCLE
    17        SEG LKPRM1,,()
    18        IN S.PRM1E
    19        SEG LKPRM2,,LKPRM1
    20        IN S.PRM2E
    21        SEG LKPRM3,,LKPRM1
    22        IN S.PRM3E
    23        SEG LKPRM4,,LKPRM1
    24        IN S.PRM4E
    25        END
```

```
DRATC:PSL.IPM-PSL(1).PRMOPAP
     1        IN S.PRMDE
     2        IN S.ALLOCATOR
     3        IN S.ACBKE,,ASBKE,,RDPKE,,ACFL9,,ASFLE
     4        IN S.RLAFE,,RLFLE,,TMRDE
     5        IN S.ERFITEM,,ALFLE,,EPCSE
     6        IN S.DAFLE,,CMFLE,,SUBJE,,CRFLE,,QYFIE
     7        IN S.PRMSE,,EQVEE
     8        IN S.ITRDE,,RDLNE,,ITRDE1,,RLPKE,,RDXXE,,STUBS1
     9        SEG LKPRM1,
    10        IN S.PRM1E
    11        SEG LKPRM2,,LKPRM1
    12        IN S.PRM2E
    13        SEG LKPRM3,,LKPRM1
    14        IN S.PRM3E
    15        SEG LKPRM4,,LKPRM1
    16        IN S.PRM4E
    17        END
```

APPENDIX C. PSLPRG ELEMENT TABLE

OMATC*PSL*IFM-PSL(1) ELEMENT TABLE

O	NAME	VERSION	TYPE
	ASFLE1		RELOCATABLE
	ASBKE		RELOCATABLE
	BUBKE		RELOCATABLE
	WRBKE		RELOCATABLE
	BLAFE		RELOCATABLE
	BLFLE		RELOCATABLE
	OBFNE		RELOCATABLE
	OBSDE		RELOCATABLE
	ITWRE1		RELOCATABLE
	ACBKE		RELOCATABLE
	UPDME		RELOCATABLE
	LLINK1		RELOCATABLE
	PRMSI		RELOCATABLE
	ERFITEM		RELOCATABLE
	OBUSE		RELOCATABLE
	ERCSF		RELOCATABLE
	FPJVE		RELOCATABLE
	BDCME		RELOCATABLE
	RDFSE		RELOCATABLE
	KTCOPL	SORTF	RELOCATABLE
	PRECOMPILER		RELOCATABLE
	ASFLE1		SYMBOLIC
	ADXXE		SYMBOLIC
	ASBKE		SYMBOLIC
	BUBKE		SYMBOLIC
	WRBKE		SYMBOLIC
	BLAFE		SYMBOLIC
	BLFLE		SYMBOLIC
	OBFNE		SYMBOLIC
	OBSDE		SYMBOLIC
	ITWRE1		SYMBOLIC
	ITRDE1		SYMBOLIC
	BDXEE		SYMBOLIC
	BLFNE		SYMBOLIC
	ACBKE		SYMBOLIC
	LLINK1		SYMBOLIC
	PRERE1		SYMBOLIC
	PRMSI		SYMBOLIC
	ERFITEM		SYMBOLIC
	OBUSE		SYMBOLIC
	ERCSF		SYMBOLIC
	FPJVE		SYMBOLIC

MTCORE	SORTF	ASM SYMB
SJOVIAL		SYMBOLIC
PRECOMPILER		ABSOLUTE
DLXEE		SYMBOLIC
DLXEE		RELOCATABLE
WRACE		SYMBOLIC
WRACE		RELOCATABLE
CHXEE		SYMBOLIC
CHXEE		RELOCATABLE
ITWRE		SYMBOLIC
ITWRE		RELOCATABLE
WRLNE		SYMBOLIC
WRLNE		RELOCATABLE
ADXEE		SYMBOLIC
ADXEE		RELOCATABLE
ACBK9		SYMBOLIC
ACBK9		RELOCATABLE
RDXXE		SYMBOLIC
RDXXE		RELOCATABLE
STUBS1		SYMBOLIC
STUBS1		RELOCATABLE
OBICE		SYMBOLIC
OBICE		RELOCATABLE
OBKWE		SYMBOLIC
OBKWE		RELOCATABLE
FDXEE		SYMBOLIC
FDXEE		RELOCATABLE
TMWRF		SYMBOLIC
TMWRF		RELOCATABLE
ITRDE		SYMBOLIC
ITRDE		RELOCATABLE
RDLNE		SYMBOLIC
ITRDE1		RELOCATABLE
RLPKE		RELOCATABLE
TMRDE		SYMBOLIC
TMRDE		RELOCATABLE
DAFLE		SYMBOLIC
DAFLF		RELOCATABLE
SRJBE		SYMBOLIC
SRJBF		RELOCATABLE
CRFLE		SYMBOLIC
CRFLE		RELOCATABLE
CHFLE		SYMBOLIC
CHFLF		RELOCATABLE
DGFLE		SYMBOLIC
DGFLE		RELOCATABLE
CLMD		SYMBOLIC
PRECOMPILER		SYMBOLIC
MD		SYMBOLIC
FS		SYMBOLIC
PSICOB		SYMBOLIC
SPFORT		SYMBOLIC
FORTRANG		SYMBOLIC
COBOLG		SYMBOLIC
COBOL		SYMBOLIC
FORTRAN		SYMBOLIC
ASM		SYMBOLIC

ASMG	SYMBOLIC
SCOBOL	SYMBOLIC
PRECBMAP	SYMBOLIC
CYFIF	SYMBOLIC
FRXXE	SYMBOLIC
ESPAE	SYMBOLIC
ITPJE	SYMBOLIC
PRSDE	SYMBOLIC
LKPGE	SYMBOLIC
PSL-STUB	SYMBOLIC
ITCLE	SYMBOLIC
ITCLE	RELOCATABLE
PULNE	RELOCATABLE
PRAUE	SYMBOLIC
PRAUE	RELOCATABLE
PFJBE	SYMBOLIC
PFJBE	RELOCATABLE
PCMLE	SYMBOLIC
PCMLE	RELOCATABLE
PRDCE	SYMBOLIC
PRDCE	RELOCATABLE
WRJBE	SYMBOLIC
WRJBE	RELOCATABLE
RPUNE	SYMBOLIC
PPUNF	RELOCATABLE
PGUNF	SYMBOLIC
PGUNE	RELOCATABLE
MAIN	SYMBOLIC
MAIN	RELOCATABLE
BLOCK	SYMBOLIC
BLOCK	RELOCATABLE
ACT1	SYMBOLIC
ACT1	RELOCATABLE
ACT2	SYMBOLIC
ACT2	RELOCATABLE
ASSIGN	SYMBOLIC
ASSIGN	RELOCATABLE
PGSCAN	SYMBOLIC
PGSCAN	RELOCATABLE
CONT	SYMBOLIC
CONT	RELOCATABLE
ENDDOP	SYMBOLIC
ENDDOP	RELOCATABLE
ENDER	SYMBOLIC
ENDER	RELOCATABLE
FULCON	SYMBOLIC
FULCON	RELOCATABLE
GENGO	SYMBOLIC
GENGO	RELOCATABLE
GENLAB	SYMBOLIC
GENLAB	RELOCATABLE
GENASS	SYMBOLIC
GENASS	RELOCATABLE
GENVAR	SYMBOLIC
GENVAR	RELOCATABLE
GETINS	SYMBOLIC
[illegible]	[illegible]

GETSTM	SYMBOLIC
GETSTM	RELOCATABLE
GOTO	SYMBOLIC
GOTO	RELOCATABLE
HOLLER	SYMBOLIC
HOLLER	RELOCATABLE
IFCASE	SYMBOLIC
IFCASE	RELOCATABLE
IFSO	SYMBOLIC
IFSO	RELOCATABLE
INDENT	SYMBOLIC
INDENT	RELOCATABLE
INITAL	SYMBOLIC
INITAL	RELOCATABLE
KLASS1	SYMBOLIC
KLASS1	RELOCATABLE
KLASS	SYMBOLIC
KLASS	RELOCATABLE
LADJUS	SYMBOLIC
LADJUS	RELOCATABLE
MOVEWD	SYMBOLIC
MOVEWD	RELOCATABLE
NDSCAN	SYMBOLIC
NDSCAN	RELOCATABLE
MOVEWU	SYMBOLIC
MOVEWU	RELOCATABLE
PUTIF	SYMBOLIC
PUTIF	RELOCATABLE
STRUCT	SYMBOLIC
STRUCT	RELOCATABLE
VERBAT	SYMBOLIC
VERBAT	RELOCATABLE
WARN	SYMBOLIC
WARN	RELOCATABLE
FILCHK	SYMBOLIC
FILCHK	RELOCATABLE
IBALPR	SYMBOLIC
IBALPR	RELOCATABLE
IGROUP	SYMBOLIC
IGROUP	RELOCATABLE
ISEND	SYMBOLIC
ISEND	RELOCATABLE
IWITHN	SYMBOLIC
IWITHN	RELOCATABLE
KEMPTY	SYMBOLIC
KEMPTY	RELOCATABLE
NAMOR	SYMBOLIC
NAMOF	RELOCATABLE
NEWLAB	SYMBOLIC
NEWLAB	RELOCATABLE
INDLEV	SYMBOLIC
INDLEV	RELOCATABLE
SPRYWD	SYMBOLIC
SPRYWD	RELOCATABLE
ACT3	SYMBOLIC
ACT3	RELOCATABLE
CPTIME	SYMBOLIC

```
CPTIME  RELOCATABLE
GETCRD  SYMBOLIC

GETCRD  RELOCATABLE
NEWPAG  SYMBOLIC

NEWPAG  RELOCATABLE
PUTIFT  SYMBOLIC

PUTIFT  RELOCATABLE
CONTRL  SYMBOLIC

CONTRL  RELOCATABLE
KCOMP   SYMBOLIC

KCOMP   RELOCATABLE
PUTFTN  SYMBOLIC

PUTFTN  RELOCATABLE
ERROR   SYMBOLIC

ERROR   RELOCATABLE
FMMDE   SYMBOLIC

PPFTMAP SYMBOLIC
EXPGE   SYMBOLIC

EXPGE   RELOCATABLE
PRMRE   SYMBOLIC

PRMRE   RELOCATABLE
CRSCE   SYMBOLIC

CRSCE   RELOCATABLE
UPMDE   SYMBOLIC

UPMDE   RELOCATABLE
MVLINE  SYMBOLIC

MVLINE  RELOCATABLE
FMMDE   RELOCATABLE

*PKLBE  RELOCATABLE
PRM4E   SYMBOLIC

PRM4E   RELOCATABLE
FMEDE   SYMBOLIC

FMEDE   RELOCATABLE
FMMVE   SYMBOLIC

FMMVE   RELOCATABLE
UPMRF   SYMBOLIC

UPMRE   RELOCATABLE
UPMFE   SYMBOLIC

UPMFE   RELOCATABLE
ITSFE   SYMBOLIC

ITSFE   RELOCATABLE
CLUNE   SYMBOLIC

CLUNE   RELOCATABLE
PCLUE   SYMBOLIC

PCLUE   RELOCATABLE
PRNIE   SYMBOLIC

PRNIE   RELOCATABLE
PRFTE   SYMBOLIC

PRFTE   RELOCATABLE
PRJVE   SYMBOLIC

PRJVE   RELOCATABLE
CLMDE   SYMBOLIC

PPCLE   SYMBOLIC
PPCLE   RELOCATABLE

PPCBF   SYMBOLIC
UPCLE   SYMBOLIC

        RELOCATABLE
```

	PPGLE	SYMBOLIC
	PRMDE	SYMBOLIC
	PRMDE	RELOCATABLE
	PRPSE	SYMBOLIC
	PRERE	SYMBOLIC
	PRERE	RELOCATABLE
	PRUNE	SYMBOLIC
	PRUNE	RELOCATABLE
	ITPJE	RELOCATABLE
	PRXXE	RELOCATABLE
	CYFIE	RELOCATABLE
	PRSDE	RELOCATABLE
	ESPAE	RELOCATABLE
	LAPGE	RELOCATABLE
	SCSGE	SYMBOLIC
	SCSGE	RELOCATABLE
	CLMDE	RELOCATABLE
	PPGLE	RELOCATABLE
	PPCBE	RELOCATABLE
	PRESE	RELOCATABLE
	PCTLE	RELOCATABLE
	CHUNE	RELOCATABLE
	CHPCE	RELOCATABLE
	CHLNE	RELOCATABLE
	CHUNE	SYMBOLIC
	CHRCE	SYMBOLIC
	CHLNE	SYMBOLIC
	CPYLIB	SYMBOLIC
	PROCLIB	COB PROC
	PRMSE	SYMBOLIC
	PRMSE	RELOCATABLE
	PRM2E	SYMBOLIC
	PRM2E	RELOCATABLE
	PRM3E	SYMBOLIC
	PRM5E	RELOCATABLE
	PRCBE	SYMBOLIC
	PRCBE	RELOCATABLE
	PRM1E	SYMBOLIC
	PPLKE	SYMBOLIC
	CPLKE	RELOCATABLE
*	ADUNE	RELOCATABLE
	PRM1E	RELOCATABLE
	PGSCE	SYMBOLIC
	PGSCE	RELOCATABLE
	PO-PCCLE	SYMBOLIC
	PCCLE	ELT SYMB
	PKSCE	SYMBOLIC
	CMMLE	SYMBOLIC
	PCCLE	RELOCATABLE
*	RSLBE	RELOCATABLE
	PKSCE	RELOCATABLE
	CMMLE	RELOCATABLE
	RUN	ELT SYMB
	PCTLMAP	ELT SYMB
	1E	SYMBOLIC
	2A	SYMBOLIC
	2I	SYMBOLIC

2C	SYMBOLIC
9A	SYMBOLIC
2B	SYMBOLIC
3C	SYMBOLIC
3D	SYMBOLIC
4	SYMBOLIC
5A	SYMBOLIC
5E	SYMBOLIC
6A	SYMBOLIC
6B	SYMBOLIC
8A	SYMBOLIC
8D	SYMBOLIC
S1	SYMBOLIC
S2	SYMBOLIC
S3	FLT SYMB
1A	SYMBOLIC
ASFLE	RELOCATABLE
ACFLE	SYMBOLIC
ACFLF	RELOCATABLE
ACFL9	RELOCATABLE
ALFLE	RELOCATABLE
ALLOCATOR	RELOCATABLE
FMUPE	SYMBOLIC
TCOMP	SYMBOLIC
FMUPE	RELOCATABLE
BCTL	ABSOLUTE
PRPSMAP	SYMBOLIC
PPGLMAP	SYMBOLIC
PPFT	ABSOLUTE
PRPS	ABSOLUTE
PPGL	ABSOLUTE
PRMDMAP	SYMBOLIC
CLMDMAP	SYMBOLIC
PPCBMAP	SYMBOLIC
PRMD	ABSOLUTE
CLMD	ABSOLUTE
PPCB	ABSOLUTE
RSLBE	SYMBOLIC
BKLBE	SYMBOLIC
ADUNE	SYMBOLIC
ALFLE	SYMBOLIC
ACFL9	SYMBOLIC
ALLOCATOR	SYMBOLIC
ASFLE	SYMBOLIC
RDCME	SYMBOLIC
BCTLE	SYMBOLIC
RDPSE	SYMBOLIC
PSL-INSTALL	FLT SYMB
PSL-INITIAL	FLT SYMB
PSL-INSTTEST	SYMBOLIC
ADUNE	RELOCATABLE
BKLBE	RELOCATABLE
RSLBE	RELOCATABLE

NEXT AVAILABLE LOCATION-

ASSEMBLER PROCEDURE TABLE EMPTY

D NAME
ACCESSMODE

ACCOUNTINGINFORMATION
ALLOCATIONTYPE

BCTLLINKNAMES
CHUNLINKNAMES

CODESFORADUN
CODESFORBKLB

CODESFORCHUN
CODESFORCRSC

CODESFORESPA
CODESFORFMSSETUPS

CODESFORITPJ
CODESFORMVUN

CODESFORPCLU
CODESFORPFJB

CODESFORPGUN
CODESFORPPRAU

CODESFORPRDC
CODESFORPRSC

CODESFORREADWRITEUNIT
CODESFORRSLB

CODESFORSTRUCTURES
CODESFORWPJB

CONTROLCARD
DEFAULTFILECLASSIFICATION
FILECONTROLFORACCESS
FILESIZES

FMSCATALOGFILESTRING
FMSOPTIONS

FMSVOLINFO
INDEXENTRY

INPUTCARD
ITPJLINKNAMES

JOBCARD
LANGUAGELINKTABLE

LENGTHOFKEYWORDVALUES
LINKNAMESFORBKSC

LINKNAMESFORDLUN
LINKNAMESFORLKPG

LINKNAMESFORPGSC
LINKNAMESFORPPRAU

LINKNAMESFORPRPUN
MESSAGELINE

MSGNBRSFORBCTL
MVUNUNITTYPEVALUES

PCLUPROCESSTYPEVALUES
PERMISSIONTYPEVALUES

PRINTUNITOPTIONDEFAULTS
PROCSTATUSFORACCESSMETHODS

PROJECTDIRECTORYENTRY
PRSDLINKNAMES

PSLCARDIDENTIFIER
PSLDATABLOCK

PSLINDEXBLOCK

D NAME
ACCESSMODEVALUES

ADUNLINKNAMES
ALLOCATIONTYPEVALUES

CHANGERECORD
CODESFORACCESSMETHODS

CODESFORBCTL
CODFSFORBKSC

CODESFORCMML
CODESFORDLUN

CODESFOREXPG
CODESFORINDEXPROCESSING

CODESFORLKPG
CODESFOROBIC

CODESFORPCML
CODFSFORPGSC

CODESFORPPLK
CODESFORPRCB

CODFSFORPRMB
CODESFORPRUN

CODESFORRPUN
CODESFORSPCHECK

CODESFORWRAC
CONDITIONSETTINGS

CRSCLINKNAMES
DEFAULTLINESPERUNIT

FILENAMES
FMSBUFFER

FMSNEWNAME
FMSPERMISSIONS

HEADERFORUNITLISTING
INITIALPARAMETERS

INPUTCARDKEYWORDVALUE
JCLCARD

KEYWORDCARD
LANGUAGETABLE

LINKNAMESFORBKLB
LINKNAMESFORCMML

LINKNAMESFORITSF
LINKNAMESFORMVUN

LINKNAMESFORPGUN
LINKNAMESFORPRMS

LINKNAMESFORRSLB
MSGDATAFORITSF

MSGNBRSFORITSF
PJRAMETERTABLE

PERMISSIONTYPE
PSINTCLASSIFICATION

PRINTUNITOPTIONS
PROGRAMSECTION

PROJECTINDEXENTRY
PRXXLINKNAMES

PSLCONTROLBLOCK
PSLHIGHERUNITBLOCK

RINCOMED

RANDOMRECORD	REQUESTEDINDEXENTRY
SCSGLINKNAMES	SECTIONOPTIONS
SECTIONTABLE	SICFSTMFLINKNAMES
SUPPORTEDLANGUAGETABLE	SYSTEMINDEXENTRY
UNITCARDRECORD	UNITLISTINGLINE
UNITLISTINGRECORD	UNITTABFRECORD
UNITTYPETABLE	UNITTYPEWORDS
USERTYPEMASTER	

FORTRAN PROCEDURE TABLE EMPTY

ENTRY POINT TABLE EMPTY